US009483347B2

(12) **United States Patent**
Wu et al.

(10) **Patent No.: US 9,483,347 B2**
(45) **Date of Patent: *Nov. 1, 2016**

(54) **SOFT-DECISION COMPENSATION FOR FLASH CHANNEL VARIATION**

(71) Applicant: **SEAGATE TECHNOLOGY LLC**, Cupertino, CA (US)

(72) Inventors: **Yunxiang Wu**, Cupertino, CA (US); **Earl T. Cohen**, Oakland, CA (US)

(73) Assignee: **Seagate Technology LLC**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/507,126**

(22) Filed: **Oct. 6, 2014**

(65) **Prior Publication Data**

US 2015/0169402 A1 Jun. 18, 2015

Related U.S. Application Data

(63) Continuation of application No. 13/567,025, filed on Aug. 4, 2012, now Pat. No. 8,856,611.

(51) **Int. Cl.**

H03M 13/00	(2006.01)
G06F 11/10	(2006.01)
G11C 29/52	(2006.01)

(52) **U.S. Cl.**

CPC ***G06F 11/1068*** (2013.01); ***G11C 29/52*** (2013.01)

(58) **Field of Classification Search**

CPC G06F 11/1068; G11C 29/52

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

8,990,665	B1 *	3/2015	Steiner et al.	714/780
2010/0088575	A1 *	4/2010	Sharon	G06F 11/1072 714/763
2012/0224420	A1 *	9/2012	Sakurada	G11C 8/08 365/185.03
2013/0219108	A1 *	8/2013	Yang	711/103

* cited by examiner

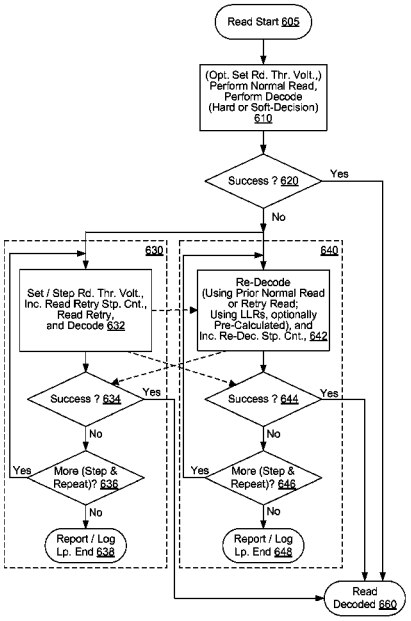
Primary Examiner — Sam Rizk

(74) *Attorney, Agent, or Firm* — Smith Tempel Blaha LLC

(57) **ABSTRACT**

In an SSD controller reading from flash memory, subsequent to failure of an initial soft-decision decoding attempt based on a nominal LLR, soft-decision re-decoding attempts are made using compensated LLR soft-decision information sets, pre-calculated at respective read-equilibrium points corresponding to mean shifts and variance change in the actual charge-state distributions of the flash memory channel. According to embodiment, soft-decision re-decoding attempts are performed without a retry read, or overlapped with one or more retry reads. By overlapping re-decoding with one or more retry reads, the probability of successful decoding increases, the need for further retry reads diminishes, and throughput is improved. The LLR compensation becomes very effective over a large number of retry reads, improving decoding reliability and achieving close to optimal bit error rates, even in the presence of large channel variation.

20 Claims, 10 Drawing Sheets



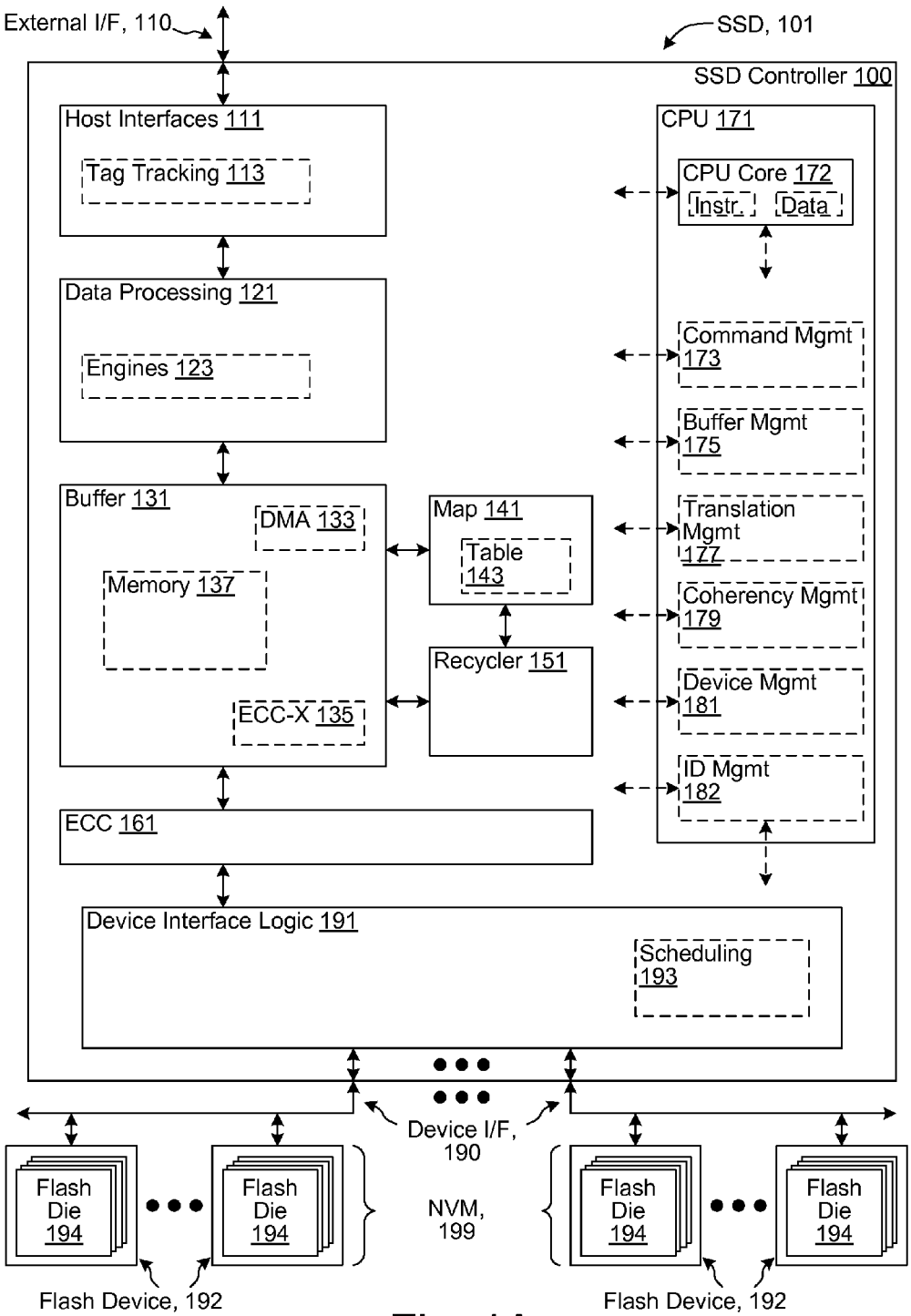

External I/F, 110
SSD, 101
SSD Controller 100
Host Interfaces 111
Tag Tracking 113
Data Processing 121
Engines 123
Buffer 131
DMA 133
Memory 137
ECC-X 135
Map 141
Table 143
Recycler 151
ECC 161
Device Interface Logic 191
Scheduling 193
CPU 171
CPU Core 172
Instr.
Data
Command Mgmt 173
Buffer Mgmt 175
Translation Mgmt 177
Coherency Mgmt 179
Device Mgmt 181
ID Mgmt 182
Device I/F, 190
NVM, 199
Flash Die 194
Flash Device, 192

Fig. 1A

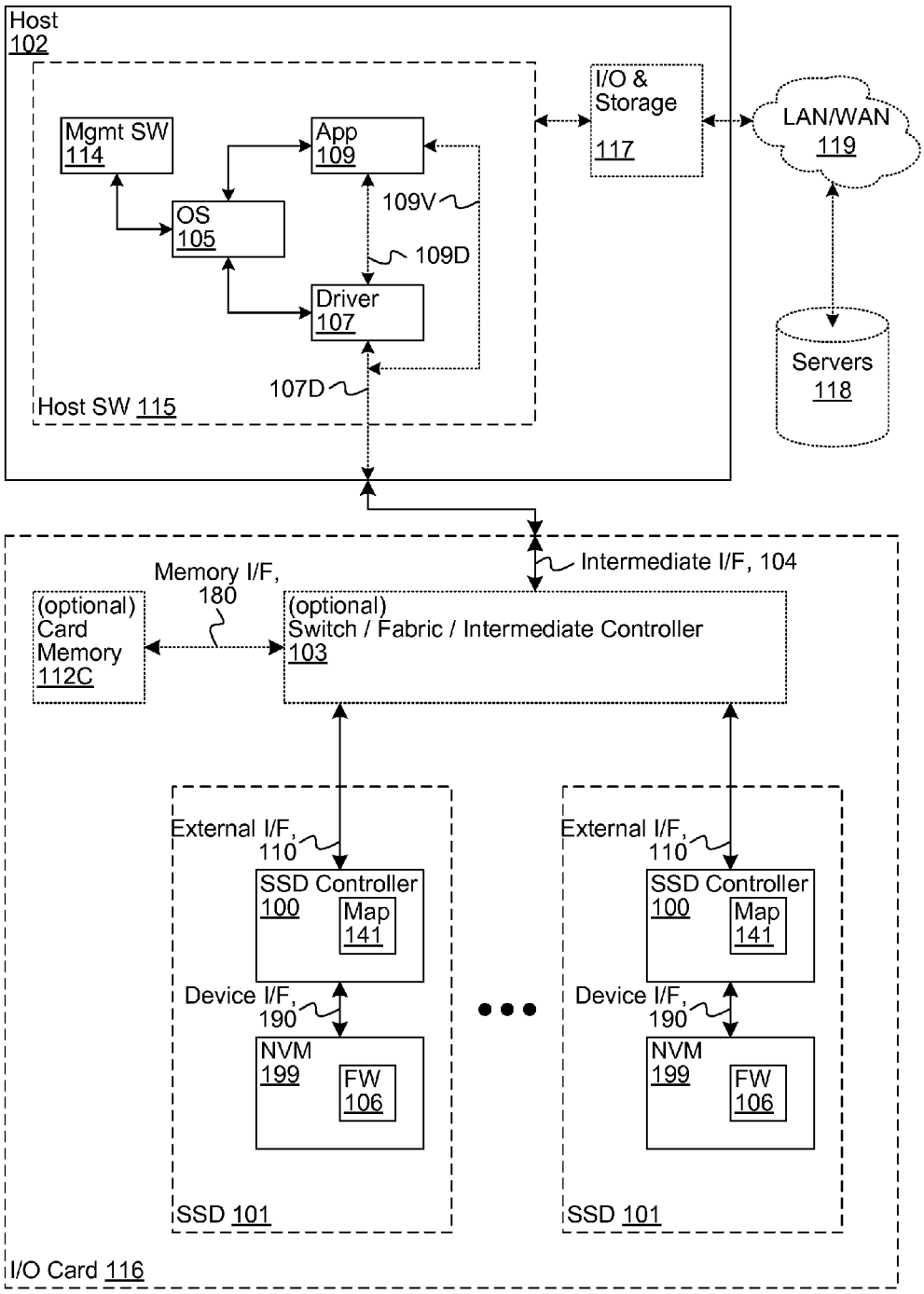

Host
102
Mgmt SW
114
App
109
OS
105
109V
109D
Driver
107
107D
Host SW 115
I/O &
Storage
117
LAN/WAN
119
Servers
118
Intermediate I/F, 104
Memory I/F,
180
(optional)
Card
Memory
112C
(optional)
Switch / Fabric / Intermediate Controller
103
External I/F,
110
SSD Controller
100
Map
141
Device I/F,
190
NVM
199
FW
106
SSD 101
External I/F,
110
SSD Controller
100
Map
141
Device I/F,
190
NVM
199
FW
106
SSD 101
I/O Card 116

Fig. 1B

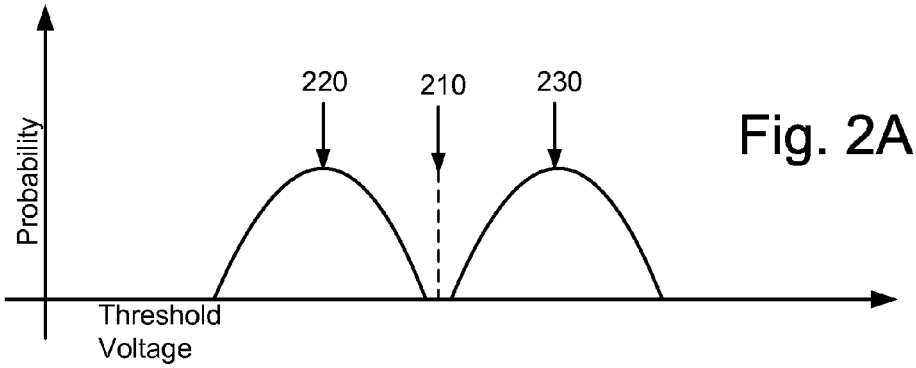


Fig. 2A

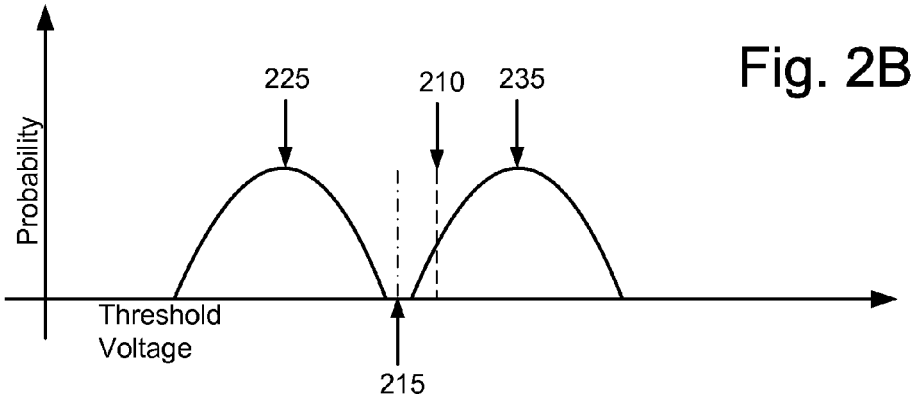


Fig. 2B

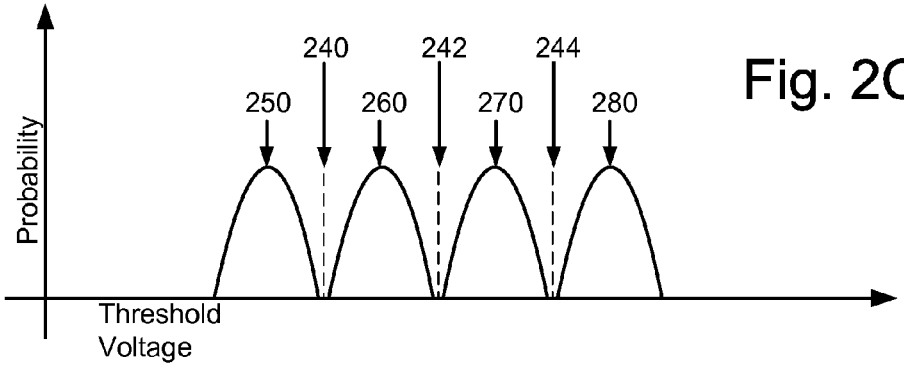


Fig. 2C

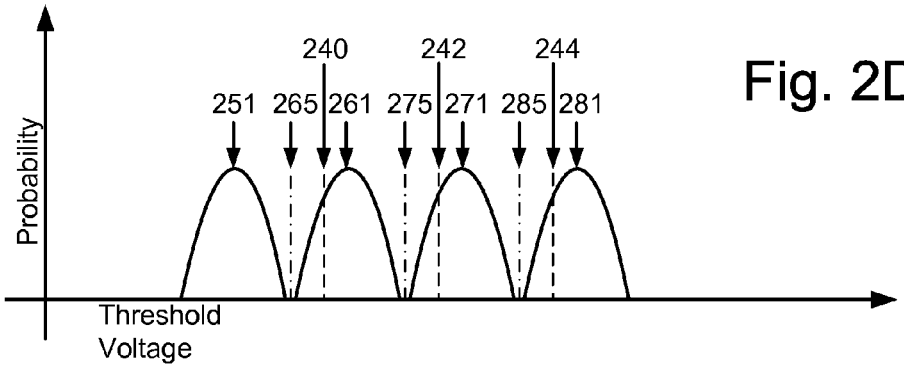


Fig. 2D

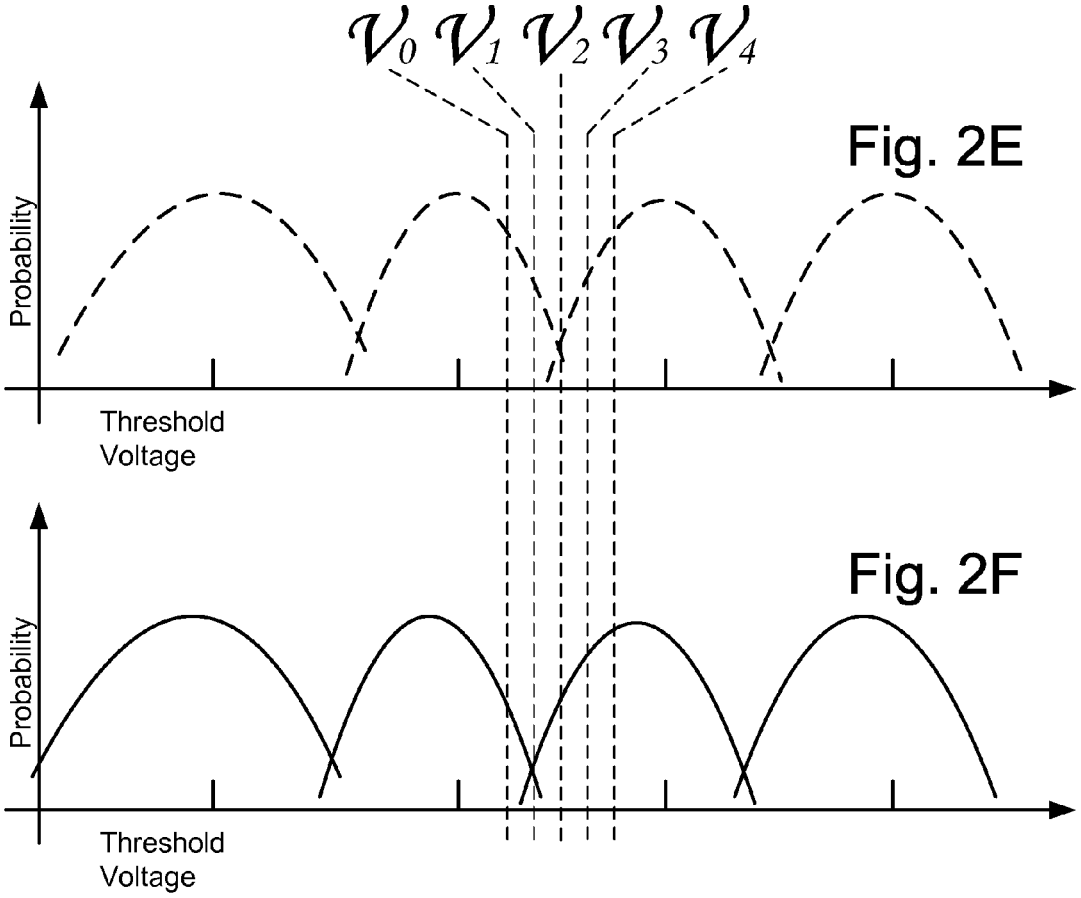

V_0 V_1 V_2 V_3 V_4
Probability
Threshold Voltage

Fig. 2E

Probability
Threshold Voltage

Fig. 2F

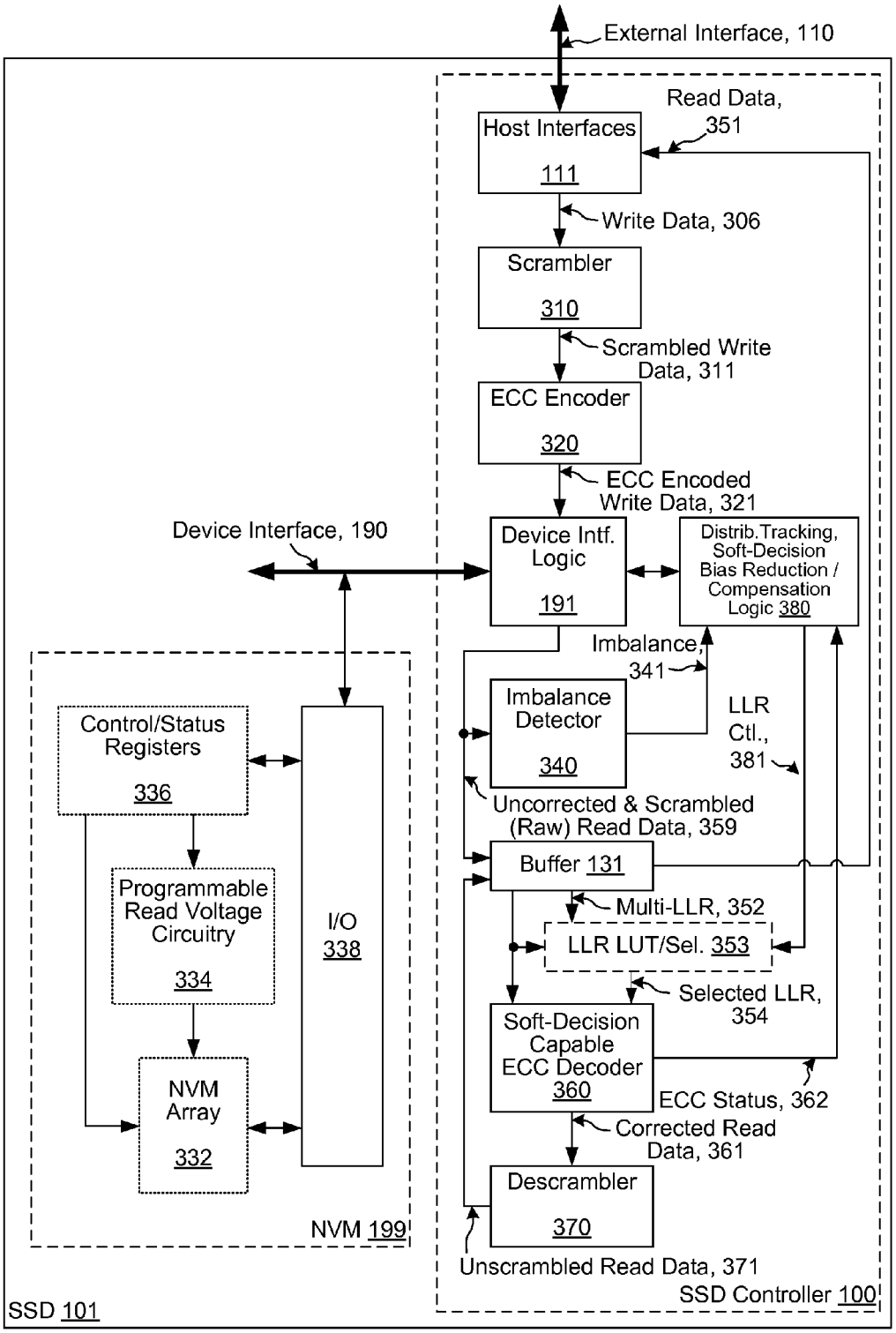

External Interface, 110
Read Data, 351
Host Interfaces 111
Write Data, 306
Scrambler 310
Scrambled Write Data, 311
ECC Encoder 320
ECC Encoded Write Data, 321
Device Interface, 190
Device Intf. Logic 191
Distrib.Tracking, Soft-Decision Bias Reduction / Compensation Logic 380
Imbalance, 341
Imbalance Detector 340
LLR Ctl., 381
Uncorrected & Scrambled (Raw) Read Data, 359
Buffer 131
Multi-LLR, 352
LLR LUT/Sel. 353
Selected LLR, 354
Soft-Decision Capable ECC Decoder 360
ECC Status, 362
Corrected Read Data, 361
Descrambler 370
Unscrambled Read Data, 371
SSD Controller 100
Control/Status Registers 336
Programmable Read Voltage Circuitry 334
NVM Array 332
I/O 338
NVM 199
SSD 101

Fig. 3A

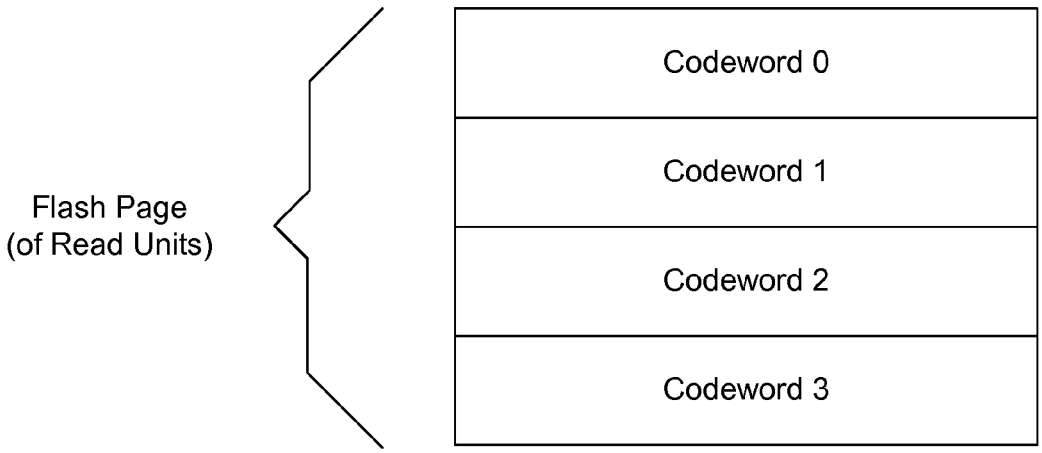

Flash Page
(of Read Units)
Codeword 0
Codeword 1
Codeword 2
Codeword 3

Fig. 3B

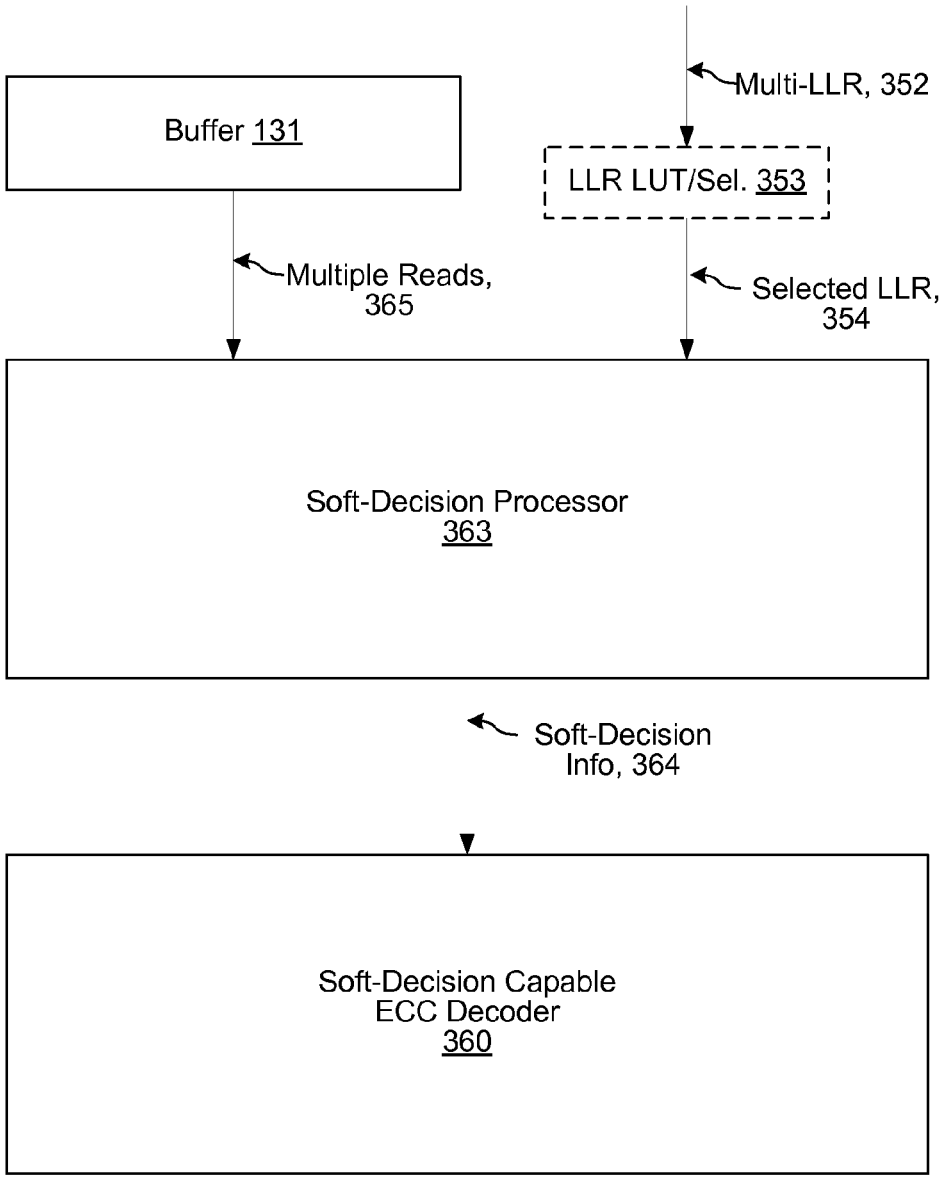

Multi-LLR, 352
Buffer 131
LLR LUT/Sel. 353
Multiple Reads, 365
Selected LLR, 354
Soft-Decision Processor 363
Soft-Decision Info, 364
Soft-Decision Capable ECC Decoder 360

Fig. 3C

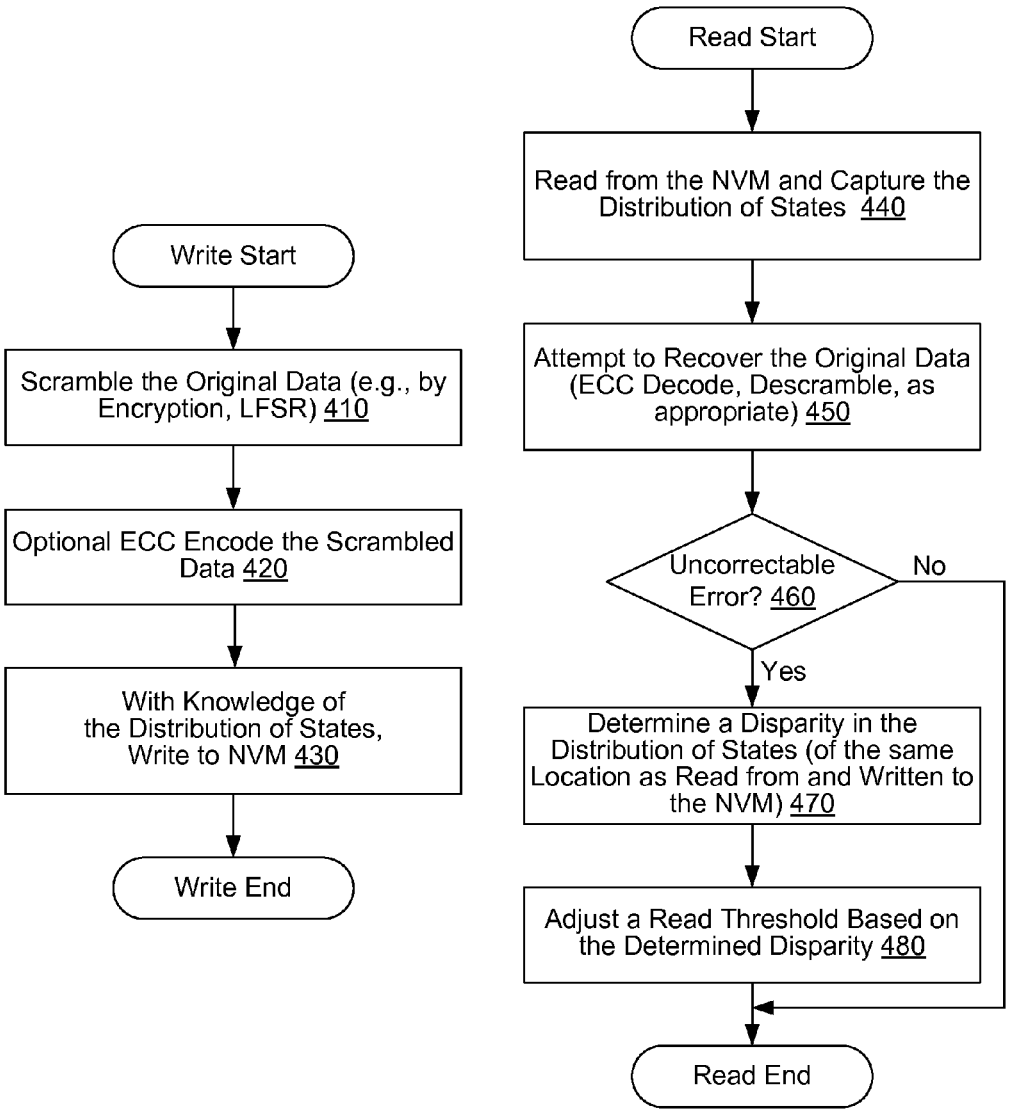

Write Start
Scramble the Original Data (e.g., by Encryption, LFSR) 410
Optional ECC Encode the Scrambled Data 420
With Knowledge of the Distribution of States, Write to NVM 430
Write End
Read Start
Read from the NVM and Capture the Distribution of States 440
Attempt to Recover the Original Data (ECC Decode, Descramble, as appropriate) 450
Uncorrectable Error? 460
No
Yes
Determine a Disparity in the Distribution of States (of the same Location as Read from and Written to the NVM) 470
Adjust a Read Threshold Based on the Determined Disparity 480
Read End

Fig. 4

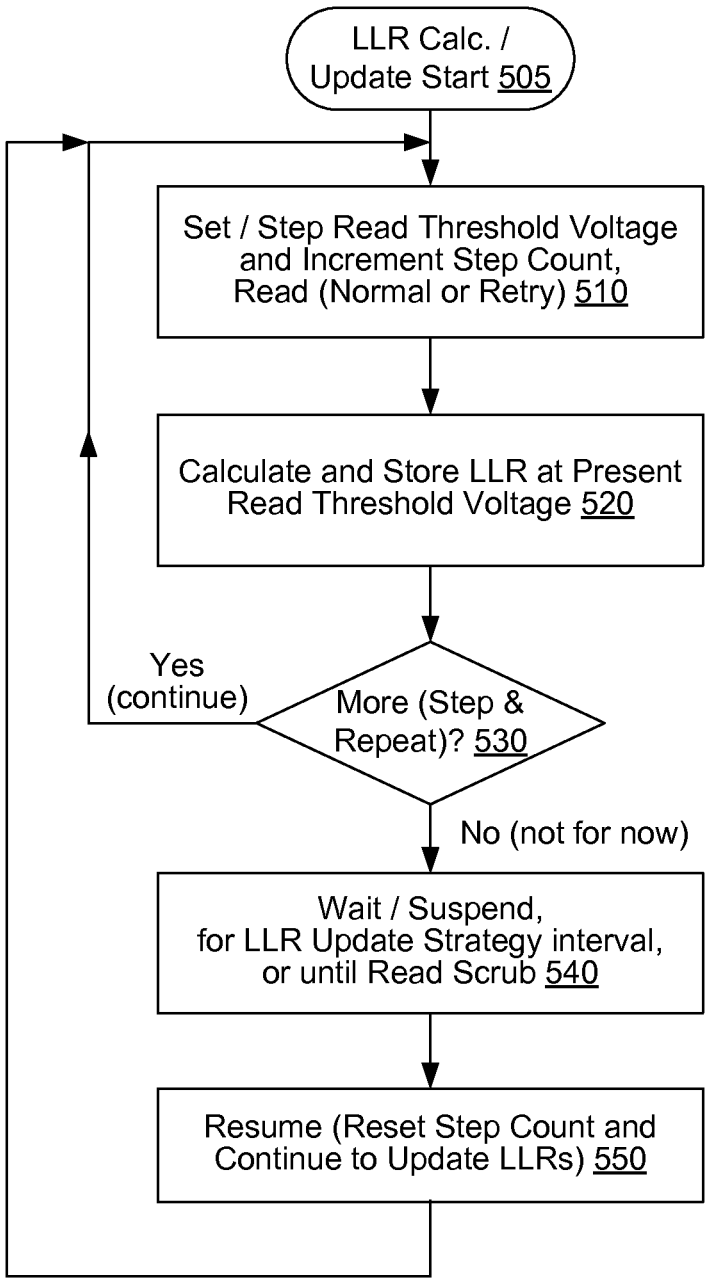

LLR Calc. / Update Start 505
Set / Step Read Threshold Voltage and Increment Step Count, Read (Normal or Retry) 510
Calculate and Store LLR at Present Read Threshold Voltage 520
More (Step & Repeat)? 530
Yes (continue)
No (not for now)
Wait / Suspend, for LLR Update Strategy interval, or until Read Scrub 540
Resume (Reset Step Count and Continue to Update LLRs) 550

Fig. 5A

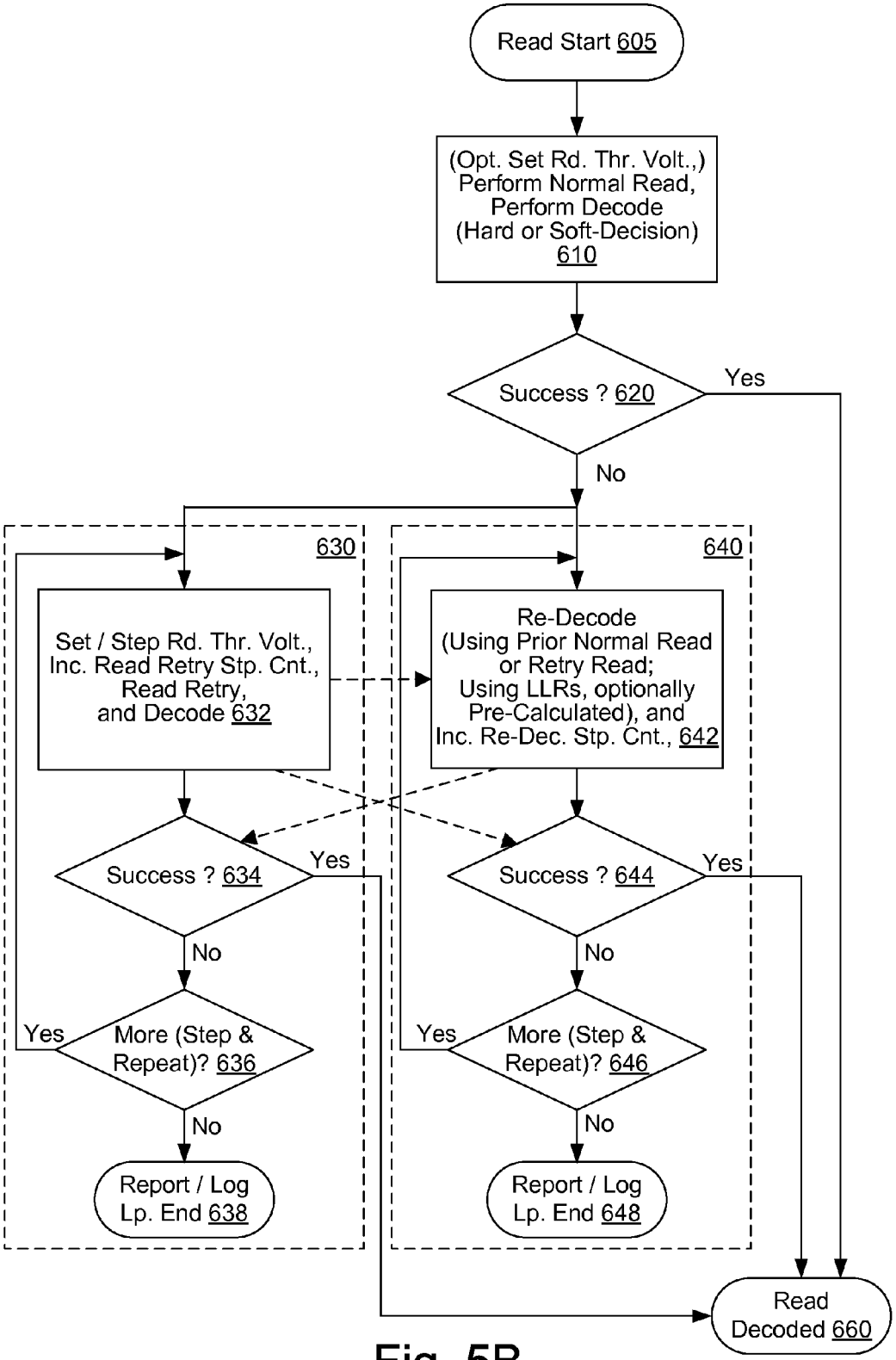

Read Start 605
(Opt. Set Rd. Thr. Volt.,) Perform Normal Read, Perform Decode (Hard or Soft-Decision) 610
Success ? 620
Yes
No
630
640
Set / Step Rd. Thr. Volt., Inc. Read Retry Stp. Cnt., Read Retry, and Decode 632
Re-Decode (Using Prior Normal Read or Retry Read; Using LLRs, optionally Pre-Calculated), and Inc. Re-Dec. Stp. Cnt., 642
Success ? 634
Yes
No
Success ? 644
Yes
No
Yes
More (Step & Repeat)? 636
No
Yes
More (Step & Repeat)? 646
No
Report / Log Lp. End 638
Report / Log Lp. End 648
Read Decoded 660

Fig. 5B

SOFT-DECISION COMPENSATION FOR FLASH CHANNEL VARIATION

CROSS REFERENCE TO RELATED APPLICATIONS

Priority benefit claims for this application are made in the accompanying Application Data Sheet. To the extent permitted by the type of the instant application, this application incorporates by reference for all purposes the following application(s), all commonly owned with the instant application at the time the invention was made:

- U.S. Non-Provisional application Ser. No. 13/567,025), filed on 4 Aug. 2012, now U.S. Pat. No. 8,856,611, first named inventor Yunxiang W U, and entitled SOFT-DECISION COMPENSATION FOR FLASH CHANNEL VARIATION;
- U.S. Non-Provisional application Ser. No. 13/567,024), filed on 4 Aug. 2012, first named inventor Yunxiang W U, and entitled SINGLE-READ BASED SOFT-DECISION DECODING OF NON-VOLATILE MEMORY;
- U.S. Non-Provisional application Ser. No. 13/533,130), filed 26 Jun. 2012, first named inventor Yingquan W U, and entitled FAST TRACKING FOR FLASH CHANNELS;
- U.S. Non-Provisional application Ser. No. 13/464,433), filed 4 May 2012, now U.S. Pat. No. 8,839,073, first named inventor Earl T COHEN, and entitled ZERO-ONE BALANCE MANAGEMENT IN A SOLID-STATE DISK CONTROLLER; and
- PCT Application and Serial No. PCT/US2011/062726), filed 30 Nov. 2011, first named inventor Jeremy WERNER, and entitled DYNAMIC HIGHER-LEVEL REDUNDANCY MODE MANAGEMENT WITH INDEPENDENT SILICON ELEMENTS.

BACKGROUND

1. Field

Advancements in non-volatile storage technology are needed to provide improvements in performance, efficiency, and utility of use.

2. Related Art

Unless expressly identified as being publicly or well known, mention herein of techniques and concepts, including for context, definitions, or comparison purposes, should not be construed as an admission that such techniques and concepts are previously publicly known or otherwise part of the prior art. All references cited herein (if any), including patents, patent applications, and publications, are hereby incorporated by reference in their entireties, whether specifically incorporated or not, for all purposes.

Various aspects of flash memory as used by Solid State Disk (SSD) controllers will now be described in part to establish a technology background and in part to establish antecedents for nomenclature used in the balance of the specification. The minimal size of data readable by the SSD controller from Non-Volatile Memory (NVM) is the "read unit", which is protected by included error correction, such as a Low-Density Parity-Check (LDPC) code. In some embodiments, each read unit contains approximately 4K to approximately 32K bits of user data, plus error correction overhead. Under command of the SSD controller, those bits are read from NVM memory cells, which depending on the technology as discussed below, may hold one or more bits per cell. In some embodiments, for security reasons an SSD controller encrypts the data prior to writing the data to NVM. In some embodiments, in view of circuit design limitations with respect to long strings of identically programmed cells, an SSD controller scrambles the data prior to writing the data to NVM.

Considered individually, each cell has a particular stored (programmed) charge that corresponds to a device threshold voltage for that cell, and further corresponds to the logical bit values being stored in the cell. While ideally all of the cells in the NVM would have identical device threshold voltages for the logical bit values stored, in practice for a variety of reasons the device threshold voltages differ across the cells in probability distributions along the threshold voltage axis (e.g., "threshold voltage distributions") that are similar to a Gaussian in shape.

Thus considered in aggregate across a large number of cells, such as of a read unit, there are as many device threshold voltage distributions (e.g., Gaussian probability curves) as there are states per cell (two states per bit of storage per cell). That is, for N-bits per cell of storage, there are 2**N states and the same number of threshold voltage distributions. Generally, (2**N)−1 different read thresholds (read voltage references V_{READ1} through $V_{READ(N-1)}$) are required by read circuits in the NVM to distinguish between the 2**N states.

Continuing from above, for Single-Level Cell (SLC) flash memories, N=1. SLC memories thus store one-bit per cell of storage, have two device threshold voltage distributions (one for zeroes and another for ones), and require a single read threshold, read voltage reference V_{READ1}. From lower to higher threshold voltages, the two threshold voltage distributions are known as the E (Erased) state and D1 (first Data) state. While arbitrary, a common mapping (coding) is to assign logical one to the E state and logical zero to the D1 state. Thus references to zeroes and ones are proxy references for respective decodings of the D1 state and the E state.

Continuing further from above, for Multi-Level Cell (MLC) flash memories, N>1. MLC memories thus store more than one bit per cell, have more than two device threshold voltage distributions, and require multiple different read thresholds to distinguish the distributions. For example, a 4LC flash memory stores 2 bits per cell, has four device threshold voltage distributions, and generally requires three read thresholds (read voltage references V_{READ1}, V_{READ2}, and V_{READ3}). From lower to higher threshold voltages, the four device threshold voltage distributions are known as the E (Erased), D1 (Data**1**), D2 (Data**2**), and D3 (Data**3**) states. While arbitrary, each of the four threshold voltage distributions is also mapped (addressed) in accordance with a particular binary sequence, such as a Gray code sequence. Thus references to one or more of the 11, 10, 00, and 01 states, are proxy references for respective decodings of the E, D1, D2, and D3 states.

With respect to address mapping used for the states of an MLC, each can be said to have a Most Significant Bit (MSB) and a Least Significant Bit (LSB) (and for more than 2 bits per cell, bits of significance in between). While there are various ways that MLC NVMs program their cells, the following approach is common. An initial programming round (a manipulation of the charge distributions) establishes the LSB, e.g., writes the "lower pages". This is done loosely in the same manner as for writing an SLC, e.g., a charge manipulation that establishes the E state threshold voltage distribution and a second state threshold voltage distribution. Depending on the binary sequence used, the second state threshold voltage distribution is similar to the D1 state threshold voltage distribution, similar to the D2

state threshold voltage distribution, or between the D1 and D2 state threshold voltage distributions. For MLC, one or more additional programming rounds further manipulate the device threshold voltage distributions (in number, location along the voltage threshold axis, and in shape), as required per the number of levels of the MLC. More particularly, one or more subsequent programming operations write the "middle pages" (if any, for more than 2 bits per cell), and a last programming operation establishes the MSB, e.g., writes the "upper pages". For example in a 4LC (2-bit per cell MLC), the E distribution and the second distribution of the first program round are respectively bifurcated by a second program round into E and D1 distributions and D2 and D3 distributions.

The threshold voltage distributions are modified away from their initial/nominal distributions by one or more factors, such as read disturb, write disturb, and retention loss. More particularly, over time, temperature, program/erase (P/E) cycling, and other factors related to use, the location of each of the threshold voltage distribution can significantly move around with respect to the threshold voltage axis. For example, over thousands of P/E cycles, retention loss can cause severe mean shifts toward lower voltage. Such changes, sometimes referred to as channel variation, increase the likelihood of read errors that are performed using a read reference voltage value for the read threshold that was previously established based on the nominal threshold voltage distribution. In some SLC embodiments, when a hard-decision uncorrectable error is encountered in a read unit read from NVM, a series of retry operations is performed to recover the read unit. The retry operations include the NVM (e.g. under direction of the SSD controller) re-reading (sampling) the read unit at different voltage values of the read threshold V_{READ1}, such as determined by a register setting written via an I/O command from the SSD controller. By reading at different settings of the read threshold V_{READ1}, the read unit is sampled at different points on the threshold voltage axis in an attempt to locate a sample of the read unit that is (hard-decision) correctable.

One algorithm suggested by NVM vendors is sweeping the SLC read threshold slowly upwards (increasing the read reference voltage V_{READ1} from its nominal value) in an attempt to find a sample of the read unit that is correctable. If this procedure fails, then the read threshold is swept in another direction (decreasing the read reference voltage V_{READ1} from its nominal value). If both sweeps fail, then the read unit is uncorrectable (by hard-decision decoding). Linearly sweeping the read threshold, which might have 16 to 64 steps at different respective read reference voltage settings, requires many time-consuming samples of the read unit (each with an attendant recovery time latency). Even when such a search is infrequently required, and thus not a major impact on average latencies, for applications with strict maximum latency requirements, including some database applications, such a time-consuming search is unacceptable.

SYNOPSIS

The invention may be implemented in numerous ways, including as a process, an article of manufacture, an apparatus, a system, a composition of matter, and a computer readable medium such as a computer readable storage medium (e.g., media in an optical and/or magnetic mass storage device such as a disk, or an integrated circuit having non-volatile storage such as flash storage) or a computer network wherein program instructions are sent over optical or electronic communication links. In this specification, these implementations, or any other form that the invention may take, may be referred to as techniques. The Detailed Description provides an exposition of one or more embodiments of the invention that enable improvements in performance, efficiency, and utility of use in the field identified above. The Detailed Description includes an Introduction to facilitate the more rapid understanding of the remainder of the Detailed Description. The Introduction includes Example Embodiments of one or more of systems, methods, articles of manufacture, and computer readable media in accordance with the concepts described herein. As is discussed in more detail in the Conclusions, the invention encompasses all possible modifications and variations within the scope of the issued claims.

The inventors believe they are the first to note the significance, with respect to read threshold management and attendant low latency performance and extended lifetime benefits, of the scrambling of data written to NVM and the tracking and managing of the statistical distribution of state of the scrambled data. More particularly, the inventors believe they are the first to note that a disparity in the statistical distributions of state of data read vs. data written is indicative of a shift in a preferred read threshold to use for reading the NVM. In some embodiments, an SSD controller scrambles data, such as via encryption, prior to writing the data to NVM. The scrambled data has a known zero/one balance (statistical distribution of bits stored as zeroes and ones), such as having close to a 50-50 balance of zero bits and one bits. With efficient scrambling, the data is effectively random. In a random sample the size of an NVM read unit, a distribution of zeroes and ones tends to be very tight (narrowly dispersed about the mean). For example, with 16K (16,384) random bits, a probability of either the zeroes count or the ones count being more than approximately 300 away from a mean (8K) in either direction is <1% (and a probability of being more than approximately 500 away in either direction is <1 in a million).

Shifts in the various threshold voltage distributions with respect to the threshold voltage axis (due to any number of factors), away from a nominal distribution, act to change the location of a center-point between the threshold voltage distribution peaks. Changes in this actual "threshold voltage center-point" without corresponding tracking adjustments to the corresponding read threshold are a cause for read errors. The inventors believe they are the first to note the significance of this change of center-point, with respect to read threshold voltage management and attendant low latency performance and extended lifetime benefits.

Exploiting the above observations, in some embodiments an SSD controller at least temporarily maintains a zero count and a one count for each read unit read from an SLC NVM (or the lower pages of an MLC). In the event that the read unit is uncorrectable in part due to the shift in the threshold voltage distributions away from their (initial) nominal distributions, the zero count and/or the one count enable a determination of a direction and/or a magnitude to move a read threshold (read voltage reference V_{READ1}) to track the shifted threshold voltage distributions and restore the zero/one balance. As presented in the Detailed Description, according to embodiment, the new voltage setting for the read threshold is determined in a variety of ways (counts, percentages) based on a number of factors (e.g. observed/inferred threshold voltage distributions, known stored values, and past NVM operating events).

The above techniques are extended for MLC memories, including maintaining counts for upper pages and shifting of

multiple read thresholds. The above techniques are also usable with a variety of encryption/scrambling methods having known zero/one balances other than a 50-50 balance.

In some embodiments of an SSD controller reading from flash memory, subsequent to failure of an initial soft-decision decoding attempt based on a nominal log-likelihood ratio (LLR), soft-decision re-decoding attempts are made using compensated LLR soft-decision information sets, pre-calculated at respective read-equilibrium points corresponding to mean shifts and variance change in the actual charge-state distributions of the flash memory channel. According to embodiment, soft-decision re-decoding attempts are performed without requiring a retry read, or overlapped with one or more retry reads. By overlapping re-decoding with one or more retry reads, the probability of successful decoding increases, the need for further retry reads diminishes, and throughput is improved. The LLR compensation becomes very effective over a large number of retry reads, improving decoding reliability and achieving close to optimal bit error rates, even in the presence of large channel variation.

BRIEF DESCRIPTION OF DRAWINGS

FIG. **1**A illustrates selected details of an embodiment of a Solid-State Disk (SSD) including an SSD controller providing a platform to implement log-likelihood ratio (LLR) compensation and zero/one balance management, in view of variations in a "channel" involving writes to and subsequent reads from non-volatile storage, such as implemented via Non-Volatile Memory (NVM) elements (e.g., flash memories).

FIG. **1**B illustrates selected details of various embodiments of systems including one or more instances of the SSD of FIG. **1**A.

FIGS. **2**A through **2**D illustrate example threshold voltage distributions relevant to zero/one balance restoration. FIGS. **2**A and **2**B are SLC specific, respectively illustrating threshold voltage distributions at a first and a later second time. FIGS. **2**C and **2**D are MLC specific, respectively illustrating threshold voltage distributions at a first and a later second time.

FIGS. **2**E and **2**F illustrate threshold voltage distributions and read threshold voltage points relevant to LLR compensation for variation in the flash memory channel. FIG. **2**E represents estimated distributions, while FIG. **2**F represents actual distributions.

FIG. **3**A illustrates selected details of a system embodiment of an instance of the SSD of FIG. **1**A, providing particular details regarding LLR compensation for flash memory channel variation and zero/one balance management.

FIG. **3**B illustrates selected details with respect to a read unit used by the embodiment of FIG. **3**A.

FIG. **3**C illustrates additional detail in some embodiments regarding a soft-decision processor that precedes or is an integral frontend of the soft-decision capable ECC decoder of FIG. **3**A.

FIG. **4** illustrates selected control flow details for an embodiment of the SSD of FIGS. **1**A, **1**B, and **3**A, providing particular details regarding write related operations (actions **410** through **430**) and read related operations (actions **440** through **480**).

FIG. **5**A illustrates an exemplary control flow for calculating, storing, and updating of LLR soft-decision information. FIG. **5**B illustrates an exemplary control flow for performing a flash memory read assisted by LLR compensation.

LIST OF REFERENCE SYMBOLS IN DRAWINGS

Ref. Symbol	Element Name
100	SSD Controller
101	SSD
102	Host
103	(optional) Switch/Fabric/Intermediate Controller
104	Intermediate Interfaces
105	OS
106	FirmWare (FW)
107	Driver
107D	dotted-arrow (Host Software ←→ I/O Device Communication)
109	Application
109D	dotted-arrow (Application ←→ I/O Device Communication via driver)
109V	dotted-arrow (Application ←→ I/O Device Communication via VF)
110	External Interfaces
111	Host Interfaces
112C	(optional) Card Memory
113	Tag Tracking
114	Multi-Device Management Software
115	Host Software
116	I/O Card
117	I/O & Storage Devices/Resources
118	Servers
119	LAN/WAN
121	Data Processing
123	Engines
131	Buffer
133	DMA
135	ECC-X
137	Memory
141	Map
143	Table
151	Recycler
161	ECC
171	CPU
172	CPU Core
173	Command Management
175	Buffer Management
177	Translation Management
179	Coherency Management
180	Memory Interface
181	Device Management
182	Identity Management
190	Device Interfaces
191	Device Interface Logic
192	Flash Device
193	Scheduling
194	Flash Die
199	NVM
210	SLC Nominal Read Threshold
215	SLC Adjusted Read Threshold
220	SLC Nominal Most-Negative Threshold State Distribution
225	SLC Shifted Most-Negative Threshold State Distribution
230	SLC Nominal Most-Positive Threshold State Distribution
235	SLC Shifted Most-Positive Threshold State Distribution
240	MLC Nominal First Read Threshold
242	MLC Nominal Second Read Threshold
244	MLC Nominal Third Read Threshold
250	MLC Nominal Most-Negative Threshold State Distribution
251	MLC Adjusted Most-Negative Threshold State Distribution
260	MLC Nominal Next-Most-Negative Threshold State Distribution
261	MLC Adjusted Next-Most-Negative Threshold State Distribution
265	MLC Adjusted First Read Threshold
270	MLC Nominal Next-Most-Positive Threshold State Distribution
271	MLC Adjusted Next-Most-Positive Threshold State Distribution
275	MLC Adjusted Second Read Threshold
280	MLC Nominal Most-Positive Threshold State Distribution
281	MLC Adjusted Most-Positive Threshold State Distribution

-continued

Ref. Symbol	Element Name
285	MLC Adjusted Third Read Threshold
306	Write Data
310	Scrambler
311	Scrambled Write Data
320	ECC Encoder
321	ECC Encoded Write Data
332	NVM Array
334	Programmable Read Voltage Circuitry
336	Control/Status Registers
338	I/O
340	Imbalance Detector
341	Imbalance
351	Read Data
352	Multi-LLR
353	LLR LUT/Selector
354	Selected LLR
359	Uncorrected & Scrambled (Raw) Read Data
360	Soft-Decision Capable ECC Decoder
361	Corrected Read Data
362	ECC Status
370	Descrambler
371	Unscrambled Read Data
380	Distribution Tracking, Soft-Decision Bias Reduction/Compensation Logic
381	LLR Control
410	Scramble Data
420	ECC Encode
430	Write NVM
440	Read NVM
450	Data Recovery
460	Uncorrectable Error Evaluation
470	Disparity Determination
480	Read Threshold Adjustment
505	LLR Calculate/Update Start
510	Set/Step Read Threshold Voltage and Increment Step Count, Read
520	Calculate and Store LLR at Present Read Threshold Voltage
530	More?
540	Wait/Suspend
550	Resume
605	Read Start
610	Perform Normal Read, Perform Decode
620	Success?
630	Action Group
632	Set/Step Rd. Thr. Volt., Inc. Read Retry Stp. Cnt., Read Retry, & Decode
634	Success?
636	More?
638	Report/Log Loop End
640	Action Group
642	Re-Decode, and Inc. Re-Decode Step Count
644	Success?
646	More?
648	Report/Log Loop End
660	Read Decoded

DETAILED DESCRIPTION

A detailed description of one or more embodiments of the invention is provided below along with accompanying figures illustrating selected details of the invention. The invention is described in connection with the embodiments. The embodiments herein are understood to be merely exemplary, the invention is expressly not limited to or by any or all of the embodiments herein, and the invention encompasses numerous alternatives, modifications, and equivalents. To avoid monotony in the exposition, a variety of word labels (including but not limited to: first, last, certain, various, further, other, particular, select, some, and notable) may be applied to separate sets of embodiments; as used herein such labels are expressly not meant to convey quality, or any form of preference or prejudice, but merely to conveniently distinguish among the separate sets. The order of some operations of disclosed processes is alterable within the scope of the invention. Wherever multiple embodiments serve to describe variations in process, method, and/or program instruction features, other embodiments are contemplated that in accordance with a predetermined or a dynamically determined criterion perform static and/or dynamic selection of one of a plurality of modes of operation corresponding respectively to a plurality of the multiple embodiments. Numerous specific details are set forth in the following description to provide a thorough understanding of the invention. The details are provided for the purpose of example and the invention may be practiced according to the claims without some or all of the details. For the purpose of clarity, technical material that is known in the technical fields related to the invention has not been described in detail so that the invention is not unnecessarily obscured.

Introduction

This introduction is included only to facilitate the more rapid understanding of the Detailed Description; the invention is not limited to the concepts presented in the introduction (including explicit examples, if any), as the paragraphs of any introduction are necessarily an abridged view of the entire subject and are not meant to be an exhaustive or restrictive description. For example, the introduction that follows provides overview information limited by space and organization to only certain embodiments. There are many other embodiments, including those to which claims will ultimately be drawn, discussed throughout the balance of the specification.

Acronyms

At least some of the various shorthand abbreviations (e.g. acronyms) defined here refer to certain elements used herein.

Acronym	Description
AES	Advanced Encryption Standard
AHCI	Advanced Host Controller Interface
API	Application Program Interface
ASCII	American Standard Code for Information Interchange
ATA	Advanced Technology Attachment (AT Attachment)
BCH	Bose Chaudhuri Hocquenghem
CD	Compact Disk
CF	Compact Flash
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DAS	Direct Attached Storage
DDR	Double-Data-Rate
DES	Data Encryption Standard
DMA	Direct Memory Access
DNA	Direct NAND Access
DRAM	Dynamic Random Access Memory
DVD	Digital Versatile/Video Disk
DVR	Digital Video Recorder
ECC	Error-Correcting Code
eMMC	Embedded MultiMediaCard
eSATA	external Serial Advanced Technology Attachment
GPS	Global Positioning System
HDD	Hard Disk Drive
I/O	Input/Output
IC	Integrated Circuit
IDE	Integrated Drive Electronics
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
LB	Logical Block
LBA	Logical Block Address
LDPC	Low-Density Parity-Check
LFSR	Linear Feedback Shift Register
LPN	Logical Page Number
LSB	Least Significant Bit
LZ	Lempel-Ziv

-continued

Acronym	Description
MLC	Multi-Level Cell
MMC	MultiMediaCard
MPEG	Moving Picture Experts Group
MSB	Most Significant Bit
NAS	Network Attached Storage
NCQ	Native Command Queuing
NVM	Non-Volatile Memory
ONA	Optimized NAND Access
ONFI	Open NAND Flash Interface
OS	Operating System
PC	Personal Computer
PCIe	Peripheral Component Interconnect express (PCI express)
PDA	Personal Digital Assistant
P/E	Program/Erase
PHY	PHYsical interface
POS	Point Of Sale
RAID	Redundant Array of Inexpensive/Independent Disks
RASIE	Redundant Array of Silicon Independent Elements
ReRAM	Resistive Random Access Memory
RS	Reed-Solomon
RSA	Rivest, Shamir & Adleman
SAN	Storage Attached Network
SAS	Serial Attached Small Computer System Interface (Serial SCSI)
SATA	Serial Advanced Technology Attachment (Serial ATA)
SCSI	Small Computer System Interface
SD	Secure Digital
SDR	Single-Data-Rate
SLC	Single-Level Cell
SMART	Self-Monitoring Analysis and Reporting Technology
SSD	Solid-State Disk/Drive
TCG	Trusted Computing Group
UFS	Unified Flash Storage
USB	Universal Serial Bus
VF	Virtual Function
WAN	Wide Area Network

Tracking and Restoring Zero/One Balances

The initial discussion that follows, of tracking and managing counts for zeroes and ones and the shifting of the read threshold used to distinguish the zero and one states, is directly applicable to SLC memories and (with minor variation, as noted) to the lower pages of MLC memories. More generally with respect to MLC memories, extension to tracking and managing counts for upper pages and the shifting of multiple read thresholds (e.g., read voltage references V_{READ1}, V_{READ2}, and V_{READ3}) is detailed in separate paragraphs below.

In some embodiments, a zero count and a one count are obtained for each read unit (or each group of read units transferred at one time from one NVM die) read from an SLC NVM. In an event that the read unit is uncorrectable in part due to the shift in the threshold voltage distributions away from their (initial) nominal distributions, the zero count and/or the one count enable a determination of a direction and/or a magnitude to move (set) a read threshold to track the shift in the threshold voltage distributions and restore the zero/one balance. The read threshold being adjusted is read voltage reference V_{READ1} for SLC memories and read voltage reference V_{READ2} for the lower pages of MLC memories. For example, in an SLC having a logical one assigned to the erase state, and written with a known statistically equal (50-50 percent, a.k.a. 50/50 percent, or simply 50/50) distribution of zeroes and ones, if a current zero/one counts indicates a predominance of ones, V_{READ1} is shifted lower; and if the current zero/one counts indicates a predominance of zeroes, V_{READ1} is shifted higher. (Substitute V_{READ2} for V_{READ1} in the foregoing example, for the lower pages of MLC memories.) A magnitude of how much to change (shift) the read threshold is optionally determined by a percentage of a number of zeroes (or ones). The magnitude is based on one or more of: the number of zeroes and/or ones observed in one or more samples; a read threshold voltage increment of the one or more samples; a knowledge of predicted threshold voltage distributions corresponding to a midpoint between the peaks of the two distributions; program/erase count (wear); read disturb counts; retention time; and other similar factors.

Finding the Read-Equilibrium Point

In further embodiments, a binary-search-like operation (iteratively sampling the same read unit at respective values of the appropriate read voltage reference) is used to find a read threshold "read-equilibrium point". This is a point on the threshold voltage axis between adjacent threshold voltage distributions resulting in raw data (pre error correction) statistical state distributions that upon reading, match within a statistical margin of error to the statistical state distributions written, or idealized models of same.

For SLC memory and random scrambling, such as resulting from encryption, the read-equilibrium point is the read threshold (read voltage reference V_{READ1}) where a zero/one balance (of the raw data read from the NVM) is most even (near identical). Under such conditions, the two state distributions have a 50-50 balance. E.g., 50% of the states read are zeroes and 50% of the states read are ones. (Substitute V_{READ2} for V_{READ1} in the foregoing, for the lower pages of MLC memories.)

In some embodiments, the read-equilibrium point corresponds to one or more of: a center point between two voltage distributions, a minima between two voltage distributions, a point at which the zero/one balance in read data is closest to 50-50, and a point determined according to interpolation of the zero/one balance found at two other points. The read equilibrium point corresponds to a center-point between the adjacent threshold voltage distributions in embodiments having symmetric adjacent threshold voltage distributions.

Finding and setting the read threshold to the desired read-equilibrium point reduces a number of reads required to find an optimal point at which to attempt hard-decision decoding. In still further embodiments, hard-decision decoding is attempted at each read threshold sampling point (each value of V_{READ1} for SLC memories, or each value of V_{READ2} for the lower pages of MLC memories) in the binary search in the event a "good enough" read threshold is found before the read threshold read-equilibrium point is found. The search is generally truncated upon finding the good enough read threshold, except as discussed below regarding possibly taking additional sampling around the determined read-equilibrium point for soft-decision decoding purposes. In various embodiments, the "good enough" read threshold results in successful hard-decision decoding of the read data. In some embodiments, a magnitude of the search steps (the read threshold voltage increment) in the binary search is determined (based), at least in part, on various factors as detailed above regarding the magnitude of the read threshold voltage change.

MLC Considerations

In MLC memories, multiple read thresholds are managed. In some embodiments this is performed by assuming uniform displacements of the multiple device-threshold-voltage distributions and changing all other read thresholds based on decisions made for the first read threshold (V_{READ1}) based on a read data zero/one balance of the lower pages.

For a 4LC NVM, theoretically there is an 11/10/00/01 balance of 25-25-25-25 (in contrast to a zero/one balance of 50-50). However, there is no single operation that directly provides this balance, given that 4LCs are generally read via

two separate read operations using the three read voltage references: V_{READ2} for lower pages and V_{READ1} and V_{READ3} for upper pages. One could thus evaluate two respective zero/one balances: between the D1 and D2 states for lower pages, and between the E and D1 states combined with between the D2 and D3 states for upper pages. Alternatively, one could carry out three separate lower-page-like reads setting the single lower-page read threshold to values near each of V_{READ1}, V_{READ2}, and V_{READ3}.

By way of example, in some 4LC NVM embodiments, read voltage references V_{READ1}, V_{READ2}, and V_{READ3} may need shifting, at least for one of the two bits stored in the cell. Working similarly to the SLC case, one of the two bits stored in the cell requires a single read at a first read threshold (V_{READ2}, in this MLC case). Ascertaining the other bit requires two additional read thresholds (V_{READ1}, and V_{READ3}, and effectively two associated respective additional reads that are performed internally by the NVM).

According to various embodiments, either: the two read thresholds (V_{READ1} and V_{READ3}) for the other bit are moved in unison by a same and/or a differing amount (under a presumption that drift is similar for both states); or the two read thresholds for the other bit are moved independently (at a cost in additional read operations). The latter option requires knowing the state of the SLC-like bit in the cell (the LSB), as the state of the SLC-like bit determines which of the two read thresholds (V_{READ1} and V_{READ3}) are used for ascertaining the corresponding other bit (the MSB).

In some 4LC embodiments, a combined "upper page" read data zero/one balance is evaluated for a simultaneous blend of the E and D1 distributions and the D2 and D3 distributions. Based on disparities in the combined zero/one balance read vs. that written, and in view of the bit sense reversal (1 to 0 for moving from E to D1, and 0 to 1 for moving from D2 to D3), the corresponding two read thresholds (V_{READ1} and V_{READ3}) are moved in unison in opposite directions. This is done such that the zero/one balance of each component of the blend moves "in the same direction" (e.g., the movements do not conflict).

In some 4LC embodiments, the movement of the two upper page read thresholds (V_{READ1} and V_{READ3}) are separately determined by respective evaluations of zero/one balance for the upper page while using a respective LSB for each bit of the read unit. When the LSB=1, the direction of movement is the opposite of that of the LSB=0 case. For example, instead of computing a disparity between a zero count and a one count in upper page read data, an upper page disparity is computed by selectively modifying each bit of upper page read data based on a corresponding bit of lower page read data, converting a sense of the bits of the upper page read data from a numeric value to a directional value where, for example, 0 represents a higher threshold voltage and 1 represents a lower threshold voltage. In some embodiments, an XNOR of the bits of the upper page read data and the corresponding bits of the lower page read data performs the converting.

Obtaining and Storing of Soft-Decision Samples

Flash devices only provide binary hard-decision information after a read operation. To enable the use of soft iterative decoding algorithms for error correction, it is desirable to convert the hard-decision information into soft-decision information. Subsequently, the soft-decision information can be provided to a soft-decision capable decoder (such as an LDPC decoder) as input.

The soft-decisions converted from a single read may not be of good enough quality for successful decoding. In some embodiments, better quality soft-decision information is obtained by reading a read unit (or each group of read units transferred at one time from one NVM die) at a variety of read threshold sampling points (for an SLC, values of V_{READ1}) about the nominal (unadjusted) read threshold to obtain multiple samples of data of the read unit, thus building up a soft-decision likelihood of a value of each bit of the read unit. A spacing of the samples depends, at least in part, on particular properties of the NVM used, such as its nominal separation of charge states.

As one example, as the read threshold voltage is increased for multiple reads of an SLC read unit (where the 0V erased stated corresponds to a logic '1'), individual bits are more likely to be sampled as a '1' than as a '0' (because the stored charge is more likely to correspond to a voltage lower than the read threshold voltage as the read threshold voltage is increased). A sequence of samples of an SLC bit ordered by increasing read threshold voltage is thus expected to have a pattern of the form 0*1*, that is zero or more zeroes followed by zero or more ones. For example, for a series of reads at respectively increasing read threshold voltages, the patterns 000, 001, 011, and 111 might be expected, and other patterns (such as 101 or 010) would be indicative of some type of error or instability. Each of the patterns corresponds to a likelihood that the SLC bit has a given value. Continuing the example, the '111' pattern indicates a "strong" one and the "000" pattern indicates a "strong" zero. The '011' pattern is closer to a one than a zero, but is not as strong a one as the '111' pattern, and the '001' pattern is closer to a zero than a one, but is not as strong a zero as the '000' pattern. Each of the patterns thus corresponds to a likelihood that the SLC bit is a zero or a one. The other (error) patterns indicate a bit whose value is unknown—having an equal likelihood of being a one or a zero. The correspondence between the patterns and the likelihood that the SLC bit is a zero or a one varies according to factors such as the values of the read threshold voltages, how closely any or a particular one of the read threshold voltages is to the read-equilibrium point, distributions of stored charges, and other factors.

As another approach in some embodiments, soft-decision information can be represented in the form of log-likelihood ratio (LLR) soft-decision information. For MLC channels, the LLR is a function of the four charge-state distributions (means and variances) and the threshold voltages for reading. Multiple sets of LLR soft-decision information are calculated corresponding to different cell voltage distribution scenarios (distributions having different means and variances). According to embodiment, the LLR calculations are performed for normal reads, a number of retry reads at varying voltages, and diagnostic or system management reads performed on demand or as a background task. The calculated LLRs are saved for subsequent use and periodically updated.

By way of example, for the estimated distributions abstracted in FIG. 2E, a set of LLR soft-decision information is calculated at each of the five indicated read voltages V_0 through V_4, including a "nominal" LLR set at V_2. Accordingly, each set of calculated LLR soft-decision information corresponds to respective distributions that are shifted (offset) from the estimated distributions by a predefined amount, and each set maps the patterns of observed samples of a given bit (ordered in increasing or decreasing read voltage threshold order) to corresponding log-likelihood ratios that the given bit is a zero or a one. For example, in some embodiments each LLR set corresponds to an assumption that the read-equilibrium point is closest to the corresponding read voltage. These pre-calculated LLR soft-decision information sets enable subsequent "compensa-

tion" for corresponding voltage shifts with respect to the estimated distributions, as described further in a section below on bias compensation. According to embodiment, the LLR soft-decision calculations are performed at 5, 10, or another number of read voltage intervals about a nominal read threshold voltage. According to embodiment, the read voltages are spaced at intervals of 0.15-volt, or another voltage offset.

In some embodiments, the LLR soft-decision information is saved as sets in one or more look-up tables. According to embodiment, the one or more look-up tables (or a cache of look-up table sets previously retrieved from the NVM) are adjacent to the ECC decoder, or appropriate look-up table sets are streamed to the ECC decoder, such as from the NVM, an on-chip SRAM (that is, an SRAM integrated with the SSD controller), or an off-chip DRAM (that is, a DRAM external to and in communications with the SSD controller). In further embodiments, a soft-decision processor operating prior to, or as an initial stage of, the ECC decoder processes the multiple reads and a selected one of the one or more look-up tables to produce soft-decision information that is used by the ECC decoder.

Bias in Soft-Decision Samples

The soft-decision information is biased, however, unless the nominal read threshold about which the samples are taken is a read threshold read-equilibrium point (as described above). If the current read threshold is not the threshold read-equilibrium point, then all the soft-decision information is biased in one direction or the other. The bias can be reduced and/or compensated for, as detailed below.

Bias Reduction for Soft-Decision Samples

To obtain unbiased (or at least less-biased) soft-decision information, a first approach is used in some embodiments. Since a zero/one balance of the read unit as sampled at each of the read threshold sampling points is readily identified, the read threshold read-equilibrium point is readily determined (according to embodiment, such as by using linear interpolation). An offset is calculated corresponding to the difference between the determined read threshold read-equilibrium point and the (old, unadjusted) nominal read threshold. Previously determined soft-decision information is enabled to be properly "re-centered" and still used by applying the offset with respect to the voltages at which the soft-decision information was previously sampled (with values larger than 1.0 or less than 0.0 being truncated). This approach requires no additional reads, but produces a differing amount of precision/accuracy for some of the data.

In some embodiments, a second approach is used to obtain less-biased soft-decision information. Since a zero/one balance of the read unit as sampled at each of the read threshold sampling points is readily identified, after collecting all required samples, the read threshold sampling point closest to the read threshold read-equilibrium point is readily determined. The closest is the read threshold sampling point with the zero/one balance most proximate to the zero/one balance written. For the SLC encryption (or scrambling) case, it is the read threshold sampling point closest to having a 50-50 balance. As all of the samples nominally have equivalent spacing from each other, the closest sample is chosen as a new nominal read threshold, and additional soft-decision information samples are optionally gathered (assuming the new nominal read threshold differs from the old). Alternatively, a binary search to find the threshold read-equilibrium point is done first, with precision of the binary search limited to a finest granularity of sampling required for the desired precision of the soft-decision information. The desired precision of the soft-decision information has an associated sample window around the new nominal read threshold. Doing a binary search as a part of multiple reads needed to obtain soft-decision information requires no additional reads, unless the old nominal read threshold falls outside the desired precision sample window.

In the SLC encryption embodiments described so far, the above techniques have focused on finding a preferred read threshold that yields a read data zero/one balance that is closest to 50-50. For cases where the threshold voltage distributions are of generally the same symmetric shape and do not substantively overlap, this preferred read threshold voltage would also correspond to the minima of the combined threshold voltage distributions.

Read Threshold Interpolated from Known Points on the Threshold Voltage Distributions

In some embodiments, another approach to finding the preferred read threshold is to instead find two points, one on each adjacent threshold voltage distribution, and determine the midpoint between these two points by interpolation. For example, sampling at the peak of each threshold voltage distribution should yield a 75/25 balance of read data zeroes/ones (or ones/zeroes, depending on which peak). Once the two peaks are identified, the calculated midpoint between the two on the threshold voltage axis is used to set a new read threshold.

If it is known a priori that the threshold voltage distributions are not uniform (e.g., asymmetrically biased on one side or the other, such as by having a long tail), then in some embodiments that information is used to interpret the position of the peak and to locate the center by a slightly more complex interpolation (e.g., not just the midpoint). Factors such as retention time and/or wear may influence the symmetry of the threshold voltage distributions and are accounted for in some embodiments. Multiple sampling points are also used in some embodiments to reveal the shape of the threshold voltage distributions.

In some embodiments where the range of threshold voltage adjustment is limited, it may not be possible to find the true peak. In particular, the threshold voltage cannot be moved less than 0V with some flash devices even though the voltage distribution of the E state extends into negative threshold voltages. Knowledge of the threshold voltage distribution still permits the midpoint to be determined via interpolation. In a first approach of some embodiments, the peak of the D1 distribution is found, and the midpoint is interpolated based on knowledge of the approximate shape of the E distribution. In a second approach of some embodiments, the point of the D1 distribution at which the read data zero/one balance is 40/60 (the opposite point from the 0V measurement) is found. The calculated midpoint between the 60/40 and the 40/60 observations is then used to set the new preferred read threshold. The calculated midpoint could be determined with greater precision with additional knowledge and/or measurement of the two threshold voltage distributions.

In some embodiments, the points being interpolated are both on a same side as the read equilibrium point. For example, knowing that a first read threshold sampling point X yields 75/25 read data zero/one balance and a second read threshold sampling point Y yields 62.5/37.5, the read equilibrium point would be close to Y+/− (X-Y) where '+' vs. '−' depends on whether Y is less than X. The direction the read equilibrium point lies is the direction corresponding to moving the read data zero/one balance closer to 50/50. In the example given, that is in the direction from X and toward Y.

Knowledge of the threshold voltage distributions would yield a more accurate interpolation rather than the simple linear interpolation indicated.

LLR Compensation of Bias in Soft-Decision Samples

To perform a read using LLR soft-decision information, the SSD controller first determines the estimated distributions in accordance with a predetermined strategy, and then sets read threshold voltages expected to work best in view of the estimated distributions. Setting the read threshold voltages affects the quality of the soft-decision decoding. In some embodiments, the thresholds are set to correspond to the crossing points of the distributions. (E.g., see V_2 re the LSB page of the estimated distributions of FIG. **2**E.) An initial "nominal" LLR set is then selected (or calculated, if not pre-calculated) based on the estimated distributions.

In some embodiments, information in the individual reads, such as the disparity (balance of 0's and 1's, given that data is scrambled and is nominally 50-50 in its 0/1 distribution) is used to select the nominal LLR set. For example, for an LSB read, the one of the reads that has the closest to 50-50 disparity is the "center" and would indicate a preferred one of the LLR sets.

Even when a zero-one balance or other distribution-tracking algorithm is used, there is no guarantee that the estimated distributions will always accurately represent the actual distributions. Accordingly, the calculated nominal LLR set may not always be optimal in view of the actual distributions, and an initial attempt based on the nominal LLR set will sometimes fail.

Recall that previously, with respect to the estimated distributions of FIG. **2**E, it was described how in some embodiments LLR soft-decision information sets are calculated at respective read voltage shifts, to enable compensation for corresponding mean shifts (and variance change, if necessary) in actual distributions. Further in this regard, FIG. **2**F represents an instance of actual distributions shifted by an exemplary amount relative to the estimated distributions of FIG. **2**E.

In some embodiments, such misestimates of the actual distributions are processed as follows. Subsequent to failure of the initial attempt based on the nominal LLR set, soft-decision re-decoding attempts are made using the other pre-calculated "compensated" LLR sets. Such re-decoding attempts eventually succeed, in some contexts, provided that the extent of shift of the actual distributions, with respect to the estimated distributions, is within the range of consideration for the pre-calculated LLR sets.

The "search" order in which the various pre-calculated LLR sets are tried varies by embodiment, including a "center" outwards ordering (alternating on either side of the center), a search downwards first and search upwards second ordering, and other predetermined orderings. For the exemplary shift of FIG. **2**F, it is expected that successful decoding will result once the LLR soft-decision set corresponding to V_1 is tried.

It is also the case that the soft-decision re-decoding will generally succeed without necessitating a retry read, which can cost 50× the time of re-decoding. According to embodiment, soft-decision re-decoding attempts are performed without a retry read, or overlapped (e.g., performed concurrently, or in parallel) with one or more retry reads. In some embodiments, if re-decoding fails while overlapped with a first number of retry reads, the process is repeated with an increased number of retry reads. By overlapping re-decoding with one or more retry reads, the probability of successful decoding increases, the need for further retry reads diminishes, and throughput is improved.

Over a large number of retry reads, LLR soft-decision information sets are calculated and stored corresponding to a wide range of read voltages. Consider an example embodiment, wherein over time there are 10 retry reads at 0.15-volt spacing. LLR soft-decision information sets are calculated and saved corresponding to each of the ten read voltages. Subsequently, there will be LLR sets at five pre-calculated read voltages well positioned about the actual distributions, for all but the extremes of the 10 retry range. A spread about the actual distributions by five soft-decision sets is significant, as it enables close to optimal bit error rates. As a result, the LLR compensation becomes very effective over the large number of retry reads, improving decoding reliability and achieving close to optimal bit error rates, even in the presence of large flash channel variation.

Example Embodiments

In concluding the introduction to the detailed description, what follows is a collection of example embodiments, including at least some explicitly enumerated as "ECs" (Example Combinations), providing additional description of a variety of embodiment types in accordance with the concepts described herein; these examples are not meant to be mutually exclusive, exhaustive, or restrictive; and the invention is not limited to these example embodiments but rather encompasses all possible modifications and variations within the scope of the issued claims and their equivalents.

EC 1. A method, comprising:

performing an initial read and an initial attempted ECC decoding, for each of a plurality of flash memory locations;

performing up to a first plurality of ECC soft-decision redecoding attempts based on the initial read, subsequent to failure of the initial attempted ECC decoding, for at least some of the locations; and

wherein each ECC soft-decision redecoding attempt uses a selected one of a plurality of log likelihood ratio (LLR) soft-decision information sets corresponding to a respective one of a plurality of read-equilibrium points.

EC 2. The method of EC 1, further comprising:

subsequent to a soft-decision-decoding success for the flash memory location, suspending further soft-decision-redecoding attempts for the flash memory location.

EC 3. The method of EC 1, further comprising:

for the at least some of the locations, overlapped in time with the first plurality of ECC soft-decision redecoding attempts, performing a second plurality of retry reads and respective retry ECC decoding attempts.

EC 4. The method of EC 1, wherein the initial attempted ECC decoding is an ECC hard-decision decoding attempt.

EC 5. The method of EC 1, wherein the initial attempted ECC decoding is an ECC soft-decision decoding attempt using an initial LLR soft-decision information set selected from the plurality of log likelihood ratio (LLR) soft-decision information sets.

EC 6. The method of EC 5, wherein at least some of the selected LLR soft-decision information sets are selected in accordance with a read threshold-tracking algorithm and a predetermined retry strategy.

EC 7. The method of EC 6, wherein the threshold-tracking algorithm comprises evaluating a zero-one balance for each of a corresponding plurality of reads.

EC 8. The method of EC 7, wherein the initial LLR soft-decision information set corresponds to an initial read

threshold voltage predicted to be associated with a lowest disparity of the zero-one balances evaluated.

EC 9. The method of EC 8, further comprising selecting subsequent LLR soft-decision information sets corresponding to read threshold voltages alternating on either side of the initial read threshold voltage in incrementally increasing amounts.

EC 10. The method of EC 1, wherein the first plurality is at least five (5) in number.

EC 11. The method of EC 1, further comprising generating and maintaining at least some of the LLR soft-decision information sets in accordance with a predetermined strategy.

EC 12. The method of EC 11, further comprising generating at least some of the LLR soft-decision information sets in advance of the soft-decision-decoding attempts.

EC 13. The method of EC 11, further comprising generating at least some of the LLR soft-decision information sets based on at least some non-retry reads and at least some retry reads.

EC 14. The method of EC 11, further comprising at least sometimes updating at least some of the LLR soft-decision information sets on a periodic basis.

EC 15. The method of EC 11, further comprising at least sometimes updating at least some of the LLR soft-decision information sets subsequent to a read scrub.

EC 16. The method of EC 11, further comprising:

wherein the LLR soft-decision information sets comprise a first one of a plurality of LLR soft-decision information groups; and

wherein at least a portion of the flash memory is organized into a plurality of redundancy groups, each of the redundancy groups being associated with a corresponding one of the LLR soft-decision information groups.

EC 17. The method of EC 16, wherein at least some of the LLR soft-decision information groups used for soft-decision-decoding and soft-decision-redecoding are specific to each of a plurality of redundancy groups in which the flash memory is organized.

EC 18. The method of EC 17, wherein data from each read and at least some of the redundancy-group-specific corresponding LLR soft-decision information groups are streamed to a soft-decision decoder.

EC 19. The method of EC 17, further comprising:

streaming data from each read and at least some of the redundancy-group-specific corresponding LLR soft-decision information groups to a soft-decision processing unit to produce soft-decision information; and

wherein the redecoding attempts are performed on the soft-decision information.

EC 20. The method of EC 11, wherein at least some of the LLR soft-decision information sets are stored in look-up tables.

EC 21. A device, comprising:

means for performing an initial read and an initial attempted ECC decoding, for each of a plurality of flash memory locations;

means for performing up to a first plurality of ECC soft-decision redecoding attempts based on the initial read, subsequent to failure of the initial attempted ECC decoding, for at least some of the locations; and

wherein each ECC soft-decision redecoding attempt uses a selected one of a plurality of log likelihood ratio (LLR) soft-decision information sets corresponding to a respective one of a plurality of read-equilibrium points.

EC 22. The device of EC 21, further comprising:

means for suspending further soft-decision-redecoding attempts for the flash memory location, subsequent to a soft-decision-decoding success for the flash memory location.

EC 23. The device of EC 21, further comprising:

means for performing a second plurality of retry reads and respective retry ECC decoding attempts, overlapped in time with the first plurality of ECC soft-decision redecoding attempts, for the at least some of the locations.

EC 24. The device of EC 21, wherein the initial attempted ECC decoding is an ECC hard-decision decoding attempt.

EC 25. The device of EC 21, wherein the initial attempted ECC decoding is an ECC soft-decision decoding attempt using an initial LLR soft-decision information set selected from the plurality of log likelihood ratio (LLR) soft-decision information sets.

EC 26. The device of EC 25, wherein at least some of the selected LLR soft-decision information sets are selected in accordance with a read threshold-tracking algorithm and a predetermined retry strategy.

EC 27. The device of EC 26, wherein the threshold-tracking algorithm comprises evaluating a zero-one balance for each of a corresponding plurality of reads.

EC 28. The device of EC 27, wherein the initial LLR soft-decision information set corresponds to an initial read threshold voltage predicted to be associated with a lowest disparity of the zero-one balances evaluated.

EC 29. The device of EC 28, further comprising means for selecting subsequent LLR soft-decision information sets corresponding to read threshold voltages alternating on either side of the initial read threshold voltage in incrementally increasing amounts.

EC 30. The device of EC 21, wherein the first plurality is at least five (5) in number.

EC 31. The device of EC 21, further comprising means for generating and maintaining at least some of the LLR soft-decision information sets in accordance with a predetermined strategy.

EC 32. The device of EC 31, further comprising means for generating at least some of the LLR soft-decision information sets in advance of the soft-decision-decoding attempts.

EC 33. The device of EC 31, further comprising means for generating at least some of the LLR soft-decision information sets based on at least some non-retry reads and at least some retry reads.

EC 34. The device of EC 31, further comprising means for at least sometimes updating at least some of the LLR soft-decision information sets on a periodic basis.

EC 35. The device of EC 31, further comprising means for at least sometimes updating at least some of the LLR soft-decision information sets subsequent to a read scrub.

EC 36. The device of EC 31, further comprising:

wherein the LLR soft-decision information sets comprise a first one of a plurality of LLR soft-decision information groups; and

wherein at least a portion of the flash memory is organized into a plurality of redundancy groups, each of the redundancy groups being associated with a corresponding one of the LLR soft-decision information groups.

EC 37. The device of EC 36, wherein at least some of the LLR soft-decision information groups used for soft-decision-decoding and soft-decision-redecoding are specific to each of a plurality of redundancy groups in which the flash memory is organized.

EC 38. The device of EC 37, wherein data from each read and at least some of the redundancy-group-specific corre-

sponding LLR soft-decision information groups are streamed to a soft-decision decoder.

EC 39. The device of EC 37, further comprising:

streaming data from each read and at least some of the redundancy-group-specific corresponding LLR soft-decision information groups to a soft-decision processing unit to produce soft-decision information; and

wherein the redecoding attempts are performed on the soft-decision information.

EC 40. The device of EC 31, wherein at least some of the LLR soft-decision information sets are stored in look-up tables.

EC 41. A flash memory read decoding method, comprising:

for each flash memory location of a plurality of flash memory locations of a flash memory, performing a first read and a respective soft-decision-decoding attempt;

subsequent to a soft-decision-decoding failure for the flash memory location, performing up to a first plurality of soft-decision-redecoding attempts based on the first read overlapped in time with one or more read retries and respective soft-decision-redecoding attempts;

subsequent to a soft-decision-decoding success for the flash memory location, suspending further soft-decision-redecoding attempts, read retries, and soft-decision-decoding attempts for the flash memory location; and

wherein each soft-decision-decoding and soft-decision-redecoding attempt uses respective log likelihood ratio (LLR) soft-decision information corresponding to a respective read-equilibrium point of a plurality of read-equilibrium points, the selection of the LLR soft-decision information used in each attempt being in accordance with a read threshold-tracking algorithm and a predetermined retry strategy.

EC 42. The method of EC 41, wherein the threshold-tracking algorithm comprises evaluating a zero-one balance for each of a corresponding plurality of reads.

EC 43. The method of EC 42, wherein an initial one of the LLR soft-decision information selected corresponds to a particular one of the read-equilibrium points predicted to be associated with a lowest disparity of the zero-one balances evaluated.

EC 44. The method of EC 43, wherein the LLR soft-decision information is selected based on a center outwards strategy.

EC 45. The method of EC 44, wherein the center outwards strategy is a strategy of using LLR soft-decision information corresponding to read-equilibrium points alternating on either side of the particular read-equilibrium point in incrementally increasing amounts.

EC 46. The method of EC 41, wherein the first plurality is five (5) in number.

EC 47. The method of EC 41, wherein the first plurality is ten (10) in number.

EC 48. The method of EC 41, further comprising:

generating at least some of the LLR soft-decision information in advance of the soft-decision-decoding attempts in accordance with a predetermined LLR generation strategy; and

maintaining the LLR soft-decision information in accordance with a predetermined LLR maintenance strategy.

EC 49. The method of EC 48, further comprising generating at least some of the LLR soft-decision information based on at least some non-retry reads and at least some retry reads.

EC 50. The method of EC 48, further comprising at least sometimes updating at least some of the LLR soft-decision information on a weekly basis.

EC 51. The method of EC 48, further comprising at least sometimes updating at least some of the LLR soft-decision information on a daily basis.

EC 52. The method of EC 48, further comprising at least sometimes updating at least some of the LLR soft-decision information subsequent to a read scrub.

EC 53. The method of E48, wherein respective portions of at least some of the LLR soft-decision information correspond to respective portions of the flash memory.

EC 54. The method of EC 48, wherein a subset of at least some of the LLR soft-decision information is specific to a subset of the flash memory.

EC 55. The method of EC 48, wherein specific sets of at least some of the LLR soft-decision information correspond respectively to specific organizational areas of the flash memory.

EC 56. The method of EC 48, wherein the flash memory is organized into a plurality of redundancy groups and at least some of the LLR soft-decision information is organized into sets respectively corresponding to each redundancy group.

EC 57. The method of EC 56, wherein at least some of the LLR soft-decision information used for soft-decision-decoding and soft-decision-redecoding is specific to each redundancy group.

EC 58. The method of EC 57, wherein data from each read and at least some of the redundancy-group-specific corresponding LLR soft-decision information are streamed to a soft-decision decoder.

EC 59. The method of EC 48, wherein at least some of the LLR soft-decision information is stored in look-up tables.

System

FIG. **1**A illustrates selected details of an embodiment of SSD **101** including an SSD controller using soft-decision compensation for channel variation for managing non-volatile storage, such as implemented via NVM elements (e.g., flash memories). SSD Controller **100** is communicatively coupled via one or more External Interfaces **110** to a host (such as host **102** of FIG. **1**B). According to various embodiments, External Interfaces **110** are one or more of: a SATA interface; a SAS interface; a PCIe interface; a Fibre Channel interface; an Ethernet Interface (such as 10 Gigabit Ethernet); a non-standard version of any of the preceding interfaces; a custom interface; or any other type of interface used to interconnect storage and/or communications and/or computing devices. For example, in some embodiments, SSD Controller **100** includes a SATA interface and a PCIe interface.

SSD Controller **100** is further communicatively coupled via one or more Device Interfaces **190** to NVM **199** including one or more storage devices, such as one or more of Flash Device **192**. According to various embodiments, Device Interfaces **190** are one or more of: an asynchronous interface; a synchronous interface; a single-data-rate (SDR) interface; a double-data-rate (DDR) interface; a DRAM-compatible DDR or DDR2 synchronous interface; an ONFI compatible interface, such as an ONFI 2.2 or ONFI 3.0 compatible interface; a Toggle-mode compatible flash interface; a non-standard version of any of the preceding interfaces; a custom interface; or any other type of interface used to connect to storage devices.

Each Flash Device **192** has, in some embodiments, one or more individual Flash Die **194**. According to type of a particular one of Flash Device **192**, a plurality of Flash Die

194 in the particular Flash Device 192 is optionally and/or selectively accessible in parallel. Flash Device 192 is merely representative of one type of storage device enabled to communicatively couple to SSD Controller 100. In various embodiments, any type of storage device is usable, such as an SLC NAND flash memory, MLC NAND flash memory, NOR flash memory, flash memory using polysilicon or silicon nitride technology-based charge storage cells, two- or three-dimensional technology-based flash memory, read-only memory, static random access memory, dynamic random access memory, ferromagnetic memory, phase-change memory, racetrack memory, ReRAM, or any other type of memory device or storage medium.

According to various embodiments, Device Interfaces 190 are organized as: one or more busses with one or more of Flash Device 192 per bus; one or more groups of busses with one or more of Flash Device 192 per bus, where busses in a group are generally accessed in parallel; or any other organization of one or more of Flash Device 192 onto Device Interfaces 190.

Continuing in FIG. 1A, SSD Controller 100 has one or more modules, such as Host Interfaces 111, Data Processing 121, Buffer 131, Map 141, Recycler 151, ECC 161, Device Interface Logic 191, and CPU 171. The specific modules and interconnections illustrated in FIG. 1A are merely representative of one embodiment, and many arrangements and interconnections of some or all of the modules, as well as additional modules not illustrated, are conceived. In a first example, in some embodiments, there are two or more Host Interfaces 111 to provide dual-porting. In a second example, in some embodiments, Data Processing 121 and/or ECC 161 are combined with Buffer 131. In a third example, in some embodiments, Host Interfaces 111 is directly coupled to Buffer 131, and Data Processing 121 optionally and/or selectively operates on data stored in Buffer 131. In a fourth example, in some embodiments, Device Interface Logic 191 is directly coupled to Buffer 131, and ECC 161 optionally and/or selectively operates on data stored in Buffer 131.

Host Interfaces 111 sends and receives commands and/or data via External Interfaces 110, and, in some embodiments, tracks progress of individual commands via Tag Tracking 113. For example, the commands include a read command specifying an address (such as an LBA) and an amount of data (such as a number of LBA quanta, e.g., sectors) to read; in response the SSD provides read status and/or read data. For another example, the commands include a write command specifying an address (such as an LBA) and an amount of data (such as a number of LBA quanta, e.g., sectors) to write; in response the SSD provides write status and/or requests write data and optionally subsequently provides write status. For yet another example, the commands include a de-allocation command (e.g. a trim command) specifying one or more addresses (such as one or more LBAs) that no longer need be allocated; in response the SSD modifies the Map accordingly and optionally provides de-allocation status. In some contexts an ATA compatible TRIM command is an exemplary de-allocation command. For yet another example, the commands include a super capacitor test command or a data hardening success query; in response, the SSD provides appropriate status. In some embodiments, Host Interfaces 111 is compatible with a SATA protocol and, using NCQ commands, is enabled to have up to 32 pending commands, each with a unique tag represented as a number from 0 to 31. In some embodiments, Tag Tracking 113 is enabled to associate an external tag for a command received via External Interfaces 110 with an internal tag used to track the command during processing by SSD Controller 100.

According to various embodiments, one or more of: Data Processing 121 optionally and/or selectively processes some or all data sent between Buffer 131 and External Interfaces 110; and Data Processing 121 optionally and/or selectively processes data stored in Buffer 131. In some embodiments, Data Processing 121 uses one or more Engines 123 to perform one or more of: formatting; reformatting; transcoding; and any other data processing and/or manipulation task.

Buffer 131 stores data sent to/from External Interfaces 110 from/to Device Interfaces 190. In some embodiments, Buffer 131 additionally stores system data, such as some or all map tables, used by SSD Controller 100 to manage one or more of Flash Device 192. In various embodiments, Buffer 131 has one or more of: Memory 137 used for temporary storage of data; DMA 133 used to control movement of data to and/or from Buffer 131; and ECC-X 135 used to provide higher-level error correction and/or redundancy functions; and other data movement and/or manipulation functions. An example of a higher-level redundancy function is a RAID-like capability, where redundancy is at a flash device (e.g., multiple ones of Flash Device 192) level and/or a flash die (e.g., Flash Die 194) level instead of at a disk level.

According to various embodiments, one or more of: ECC 161 optionally and/or selectively processes some or all data sent between Buffer 131 and Device Interfaces 190; and ECC 161 optionally and/or selectively processes data stored in Buffer 131. In some embodiments, ECC 161 is used to provide lower-level error correction and/or redundancy functions, such as in accordance with one or more ECC techniques. In some embodiments, ECC 161 implements one or more of: a CRC code; a Hamming code; an RS code; a BCH code; an LDPC code; a Viterbi code; a trellis code; a hard-decision code; a soft-decision code; an erasure-based code; any error detecting and/or correcting code; and any combination of the preceding. In some embodiments, ECC 161 includes one or more decoders (such as LDPC decoders).

Device Interface Logic 191 controls instances of Flash Device 192 via Device Interfaces 190. Device Interface Logic 191 is enabled to send data to/from the instances of Flash Device 192 according to a protocol of Flash Device 192. Device Interface Logic 191 includes Scheduling 193 to selectively sequence control of the instances of Flash Device 192 via Device Interfaces 190. For example, in some embodiments, Scheduling 193 is enabled to queue operations to the instances of Flash Device 192, and to selectively send the operations to individual ones of the instances of Flash Device 192 (or Flash Die 194) as individual ones of the instances of Flash Device 192 (or Flash Die 194) are available.

Map 141 converts between data addressing used on External Interfaces 110 and data addressing used on Device Interfaces 190, using Table 143 to map external data addresses to locations in NVM 199. For example, in some embodiments, Map 141 converts LBAs used on External Interfaces 110 to block and/or page addresses targeting one or more Flash Die 194, via mapping provided by Table 143. For LBAs that have never been written since drive manufacture or de-allocation, the Map points to a default value to return if the LBAs are read. For example, when processing a de-allocation command, the Map is modified so that entries corresponding to the de-allocated LBAs point to one of the default values. In various embodiments, there are various default values, each having a corresponding pointer. The plurality of default values enables reading some de-allocated LBAs (such as in a first range) as one default value, while

reading other de-allocated LBAs (such as in a second range) as another default value. The default values, in various embodiments, are defined by flash memory, hardware, firmware, command and/or primitive arguments and/or parameters, programmable registers, or various combinations thereof.

In some embodiments, Map **141** uses Table **143** to perform and/or to look up translations between addresses used on External Interfaces **110** and data addressing used on Device Interfaces **190**. According to various embodiments, Table **143** is one or more of: a one-level map; a two-level map; a multi-level map; a map cache; a compressed map; any type of mapping from one address space to another; and any combination of the foregoing. According to various embodiments, Table **143** includes one or more of: static random access memory; dynamic random access memory; NVM (such as flash memory); cache memory; on-chip memory; off-chip memory; and any combination of the foregoing.

In some embodiments, Recycler **151** performs garbage collection. For example, in some embodiments, instances of Flash Device **192** contain blocks that must be erased before the blocks are re-writeable. Recycler **151** is enabled to determine which portions of the instances of Flash Device **192** are actively in use (e.g., allocated instead of de-allocated), such as by scanning a map maintained by Map **141**, and to make unused (e.g., de-allocated) portions of the instances of Flash Device **192** available for writing by erasing them. In further embodiments, Recycler **151** is enabled to move data stored within instances of Flash Device **192** to make larger contiguous portions of the instances of Flash Device **192** available for writing.

In some embodiments, instances of Flash Device **192** are selectively and/or dynamically configured, managed, and/or used to have one or more bands for storing data of different types and/or properties. A number, arrangement, size, and type of the bands are dynamically changeable. For example, data from a computing host is written into a hot (active) band, while data from Recycler **151** is written into a cold (less active) band. In some usage scenarios, if the computing host writes a long, sequential stream, then a size of the hot band grows, whereas if the computing host does random writes or few writes, then a size of the cold band grows.

CPU **171** controls various portions of SSD Controller **100**. CPU **171** includes CPU Core **172**. CPU Core **172** is, according to various embodiments, one or more single-core or multi-core processors. The individual processors cores in CPU Core **172** are, in some embodiments, multi-threaded. CPU Core **172** includes instruction and/or data caches and/or memories. For example, the instruction memory contains instructions to enable CPU Core **172** to execute programs (e.g. software sometimes called firmware) to control SSD Controller **100**. In some embodiments, some or all of the firmware executed by CPU Core **172** is stored on instances of Flash Device **192** (as illustrated, e.g., as Firmware **106** of NVM **199** in FIG. **1**B).

In various embodiments, CPU **171** further includes: Command Management **173** to track and control commands received via External Interfaces **110** while the commands are in progress; Buffer Management **175** to control allocation and use of Buffer **131**; Translation Management **177** to control Map **141**; Coherency Management **179** to control consistency of data addressing and to avoid conflicts such as between external data accesses and recycle data accesses; Device Management **181** to control Device Interface Logic **191**; Identity Management **182** to control modification and communication of identify information, and optionally other management units. None, any, or all of the management functions performed by CPU **171** are, according to various embodiments, controlled and/or managed by hardware, by software (such as firmware executing on CPU Core **172** or on a host connected via External Interfaces **110**), or any combination thereof.

In some embodiments, CPU **171** is enabled to perform other management tasks, such as one or more of: gathering and/or reporting performance statistics; implementing SMART; controlling power sequencing, controlling and/or monitoring and/or adjusting power consumption; responding to power failures; controlling and/or monitoring and/or adjusting clock rates; and other management tasks.

Various embodiments include a computing-host flash memory controller that is similar to SSD Controller **100** and is compatible with operation with various computing hosts, such as via adaptation of Host Interfaces **111** and/or External Interfaces **110**. The various computing hosts include one or any combination of a computer, a workstation computer, a server computer, a storage server, a SAN, a NAS device, a DAS device, a storage appliance, a PC, a laptop computer, a notebook computer, a netbook computer, a tablet device or computer, an ultrabook computer, an electronic reading device (such as an e-reader), a PDA, a navigation system, a (handheld) GPS device, an automotive control system, an automotive media control system or computer, a printer, copier or fax machine or all-in-one device, a POS device, a cash-register, a media player, a television, a media recorder, a DVR, a digital camera, a cellular handset, a cordless telephone handset, and an electronic game.

In various embodiments, all or any portions of an SSD controller (or a computing-host flash memory controller) are implemented on a single IC, a single die of a multi-die IC, a plurality of dice of a multi-die IC, or a plurality of ICs. For example, Buffer **131** is implemented on a same die as other elements of SSD Controller **100**. For another example, Buffer **131** is implemented on a different die than other elements of SSD Controller **100**.

FIG. **1**B illustrates selected details of various embodiments of systems including one or more instances of the SSD of FIG. **1**A. SSD **101** includes SSD Controller **100** coupled to NVM **199** via Device Interfaces **190**. The figure illustrates various classes of embodiments: a single SSD coupled directly to a host, a plurality of SSDs each respectively coupled directly to a host via respective external interfaces, and one or more SSDs coupled indirectly to a host via various interconnection elements.

As an example embodiment of a single SSD coupled directly to a host, one instance of SSD **101** is coupled directly to Host **102** via External Interfaces **110** (e.g. Switch/Fabric/Intermediate Controller **103** is omitted, bypassed, or passed-through). As an example embodiment of a plurality of SSDs each coupled directly to a host via respective external interfaces, each of a plurality of instances of SSD **101** is respectively coupled directly to Host **102** via a respective instance of External Interfaces **110** (e.g. Switch/Fabric/Intermediate Controller **103** is omitted, bypassed, or passed-through). As an example embodiment of one or more SSDs coupled indirectly to a host via various interconnection elements, each of one or more instances of SSD **101** is respectively coupled indirectly to Host **102**. Each indirect coupling is via a respective instance of External Interfaces **110** coupled to Switch/Fabric/Intermediate Controller **103**, and Intermediate Interfaces **104** coupling to Host **102**.

Some of the embodiments including Switch/Fabric/Intermediate Controller **103** also include Card Memory **112**C coupled via Memory Interface **180** and accessible by the

SSDs. In various embodiments, one or more of the SSDs, the Switch/Fabric/Intermediate Controller, and/or the Card Memory are included on a physically identifiable module, card, or pluggable element (e.g. I/O Card **116**). In some embodiments, SSD **101** (or variations thereof) corresponds to a SAS drive or a SATA drive that is coupled to an initiator operating as Host **102**.

Host **102** is enabled to execute various elements of Host Software **115**, such as various combinations of OS **105**, Driver **107**, Application **109**, and Multi-Device Management Software **114**. Dotted-arrow **107**D is representative of Host Software ← → I/O Device Communication, e.g. data sent/received to/from one or more of the instances of SSD **101** and from/to any one or more of OS **105** via Driver **107**, Driver **107**, and Application **109**, either via Driver **107**, or directly as a VF.

OS **105** includes and/or is enabled to operate with drivers (illustrated conceptually by Driver **107**) for interfacing with the SSD. Various versions of Windows (e.g. 95, 98, ME, NT, XP, 2000, Server, Vista, and 7), various versions of Linux (e.g. Red Hat, Debian, and Ubuntu), and various versions of MacOS (e.g. 8, 9 and X) are examples of OS **105**. In various embodiments, the drivers are standard and/or generic drivers (sometimes termed "shrink-wrapped" or "pre-installed") operable with a standard interface and/or protocol such as SATA, AHCI, or NVM Express, or are optionally customized and/or vendor specific to enable use of commands specific to SSD **101**. Some drives and/or drivers have pass-through modes to enable application-level programs, such as Application **109** via Optimized NAND Access (sometimes termed ONA) or Direct NAND Access (sometimes termed DNA) techniques, to communicate commands directly to SSD **101**, enabling a customized application to use commands specific to SSD **101** even with a generic driver. ONA techniques include one or more of: use of non-standard modifiers (hints); use of vendor-specific commands; communication of non-standard statistics, such as actual NVM usage according to compressibility; and other techniques. DNA techniques include one or more of: use of non-standard commands or vendor-specific providing unmapped read, write, and/or erase access to the NVM; use of non-standard or vendor-specific commands providing more direct access to the NVM, such as by bypassing formatting of data that the I/O device would otherwise do; and other techniques. Examples of the driver are a driver without ONA or DNA support, an ONA-enabled driver, a DNA-enabled driver, and an ONA/DNA-enabled driver. Further examples of the driver are a vendor-provided, vendor-developed, and/or vendor-enhanced driver, and a client-provided, client-developed, and/or client-enhanced driver.

Examples of the application-level programs are an application without ONA or DNA support, an ONA-enabled application, a DNA-enabled application, and an ONA/DNA-enabled application. Dotted-arrow **109**D is representative of Application ← → I/O Device Communication (e.g. bypass via a driver or bypass via a VF for an application), e.g. an ONA-enabled application and an ONA-enabled driver communicating with an SSD, such as without the application using the OS as an intermediary. Dotted-arrow **109**V is representative of Application ← → I/O Device Communication (e.g. bypass via a VF for an application), e.g. a DNA-enabled application and a DNA-enabled driver communicating with an SSD, such as without the application using the OS or the driver as intermediaries.

One or more portions of NVM **199** are used, in some embodiments, for firmware storage, e.g. Firmware **106**. The firmware storage includes one or more firmware images (or portions thereof). A firmware image has, for example, one or more images of firmware, executed, e.g., by CPU Core **172** of SSD Controller **100**. A firmware image has, for another example, one or more images of constants, parameter values, and NVM device information, referenced, e.g. by the CPU core during the firmware execution. The images of firmware correspond, e.g., to a current firmware image and zero or more previous (with respect to firmware updates) firmware images. In various embodiments, the firmware provides for generic, standard, ONA, and/or DNA operating modes. In some embodiments, one or more of the firmware operating modes are enabled (e.g. one or more APIs are "unlocked") via keys or various software techniques, optionally communicated and/or provided by a driver.

In some embodiments lacking the Switch/Fabric/Intermediate Controller, the SSD is coupled to the Host directly via External Interfaces **110**. In various embodiments, SSD Controller **100** is coupled to the Host via one or more intermediate levels of other controllers, such as a RAID controller. In some embodiments, SSD **101** (or variations thereof) corresponds to a SAS drive or a SATA drive and Switch/Fabric/Intermediate Controller **103** corresponds to an expander that is in turn coupled to an initiator, or alternatively Switch/Fabric/Intermediate Controller **103** corresponds to a bridge that is indirectly coupled to an initiator via an expander. In some embodiments, Switch/Fabric/Intermediate Controller **103** includes one or more PCIe switches and/or fabrics.

In various embodiments, such as some of the embodiments where Host **102** is a computing host (e.g. a computer, a workstation computer, a server computer, a storage server, a SAN, a NAS device, a DAS device, a storage appliance, a PC, a laptop computer, a notebook computer, and/or a netbook computer), the computing host is optionally enabled to communicate (e.g. via optional I/O & Storage Devices/Resources **117** and optional LAN/WAN **119**) with one or more local and/or remote servers (e.g. optional Servers **118**). The communication enables, for example, local and/or remote access, management, and/or usage of any one or more of SSD **101** elements. In some embodiments, the communication is wholly or partially via Ethernet. In some embodiments, the communication is wholly or partially via Fibre Channel. LAN/WAN **119** is representative, in various embodiments, of one or more Local and/or Wide Area Networks, such as any one or more of a network in a server farm, a network coupling server farms, a metro-area network, and the Internet.

In various embodiments, an SSD controller and/or a computing-host flash memory controller in combination with one or more NVMs are implemented as a non-volatile storage component, such as a USB storage component, a CF storage component, an MMC storage component, an eMMC storage component, a Thunderbolt storage component, a UFS storage component, an SD storage component, a Memory Stick storage component, and an xD-picture card storage component.

In various embodiments, all or any portions of an SSD controller (or a computing-host flash memory controller), or functions thereof, are implemented in a host that the controller is to be coupled with (e.g., Host **102** of FIG. **1**B). In various embodiments, all or any portions of an SSD controller (or a computing-host flash memory controller), or functions thereof, are implemented via hardware (e.g., logic circuitry), software and/or firmware (e.g., driver software or SSD control firmware), or any combination thereof. For example, functionality of or associated with an ECC unit (such as similar to ECC **161** and/or ECC-X **135** of FIG. **1**A)

is implemented partially via software on a host and partially via a combination of firmware and hardware in an SSD controller. For another example, functionality of or associated with a recycler unit (such as similar to Recycler **151** of FIG. **1**A) is implemented partially via software on a host and partially via hardware in a computing-host flash memory controller.

Zero/One Balance Restoration

FIGS. **2**A through **2**D illustrate threshold voltage distributions relevant to "zero/one" balance restoration. For the purposes of this discussion, each hill-like bump is an abstraction for an independent Gaussian-like curve representing a threshold voltage probability distribution for a respective state of a read unit sized portion of an NVM. That is, the number of devices (a.k.a., cells, transistors) that the distribution represents is given by the number of bits in the read unit divided by the number of bits being stored per device. The threshold voltage axis is drawn with increasing positive voltage to the right. An absolute scale is purposefully not provided, and no reference points are identified, such that the plots apply more generally to a larger population of NVMs. FIGS. **2**A and **2**B are SLC specific, while FIGS. **2**C and **2**D are MLC specific.

In FIG. **2**A, representing an initial time (e.g., when the corresponding data is written), leftmost distribution **220** represents a logical one and the rightmost distribution **230** represents a logical zero. Depending on the technology, the leftmost distribution may reside (at least predominantly) at negative voltages. Read threshold **210** is ideally situated between the two distributions.

Except as linked by the embodiments herein, independent of these threshold voltage distributions, in some embodiments, the NVM is written with specific knowledge of the statistical distribution of zeroes and ones being stored. More particularly, in some embodiments various encryption and/or scrambling techniques are used such that the statistical distribution of zeroes and ones is 50-50 percent (50 percent zeroes and 50 percent ones). When the SLC is read using the nominal read threshold, which is ideally situated for the case of FIG. **2**A, the observed read data statistical distribution of zeroes and ones is likewise 50-50 percent. The statistical distribution being 50-50 percent does not mean that any one sample of data would have an exactly even balance of zero bits and one bits, but rather that an average over many samples produces a ratio of zero bits and one bits that converges on 50-50 percent with an increasingly tight probabilistic bound as a number of the samples increases. An analogy is a distribution of heads and tails when a coin is flipped many times, generating a Gaussian distribution with a variance that is approximated by n/4 where n is a number of the coin flips. For example, if there are 18,432 bits in a read unit with a 50-50 percent statistical distribution of zero bits and one bits, the variance in the number of zero bits (or one bits) is approximately 4,608 and the standard deviation is approximately 68. With a standard deviation of 68, less than 1 in one million samples of read units would be expected to have a number of zero bits more than 340 (5 standard deviations) away from the average of 9,216.

In FIG. **2**B, representing a later time, nominal read threshold **210** is as in FIG. **2**A, and the two threshold voltage distributions **225** and **235** are shifted with respect to their earlier respective distributions **220** and **230** in FIG. **2**A. For the purposes of example, the two distributions are shown as having both uniformly shifted to the left (toward more negative voltages). It should be understood that more generally the two distributions are enabled to move independently of each other and in either a positive or negative direction.

In view of FIG. **2**B, when the SLC is again read using nominal read threshold **210**, it is predictable that the observed statistical distribution of zeroes and ones read directly from the NVM (e.g., prior to any error correction) will not be 50-50 percent. More particularly, for the conceptual example given, an erroneous excess of ones is to be expected, as the read threshold is such that some of the zeroes will be falsely read as ones.

In practice, the direction of inference is reversed. That is, in practice, generally such shifts in the threshold voltage distributions are not known or directly knowable. In some embodiments, instead the observation of a disparity in the zeroes and ones read from the NVM (with respect to the known stored distribution of zeroes and ones) is used to infer the existence of shifts in the threshold voltage distributions. Furthermore, in these embodiments the read threshold is adjusted based at least in part on the disparity observed (as detailed elsewhere herein) to read threshold **215**, as required until the zero/one balance is restored.

Similarly, FIG. **2**C represents an initial time (such as when the corresponding data is written) with initial threshold voltage distributions, while FIG. **2**D represents a later time with corresponding later threshold voltage distributions. More specifically, for a selected Gray code mapping, respectively representing the 11, 10, 00, and 10 states, threshold voltage distributions **251**, **261**, **271**, and **281** (of FIG. **2**D) are shifted with respect to their earlier respective threshold voltage distributions **250**, **260**, **270**, and **280** (of FIG. **2**C). Three nominal (initial) read thresholds are also shown: V_{READ1} **240**, V_{READ2} **242**, and V_{READ3} **244**. Again for the purposes of example, in FIG. **2**D the four distributions are shown as having all uniformly shifted to the left (toward more negative voltages). It should be understood that more generally the four distributions are enabled to move independently of each other and in either a positive or negative direction.

In some embodiments, various scrambling techniques are used such that the statistical distribution of the four states is 25-25-25-25 percent (25 percent in each state). When the MLC is read using the nominal read thresholds, which are ideally situated for the case of FIG. **2**C, in some embodiments the statistical distribution of the four states is enabled to be confirmed to likewise be 25-25-25-25 percent. (As discussed elsewhere, zeroes and ones are directly observable by the controller, but all four states are inferable as required.) In some embodiments, the observation of a disparity (a deviation from the expected 25-25-25-25 percent) read from the NVM (with respect to the known stored distribution of states) is used to infer the existence of shifts in the threshold voltage distributions. The read thresholds are then adjusted (as detailed elsewhere herein) as shown in FIG. **2**D to become V_{READ1} **265**, V_{READ2} **275**, and V_{READ3} **285**. In some embodiments, the read threshold adjustments are performed separately for lower page reads (adjusting V_{READ2} **275**) and upper page reads (adjusting V_{READ1} **265** and/or V_{READ3} **285**).

Functions for Zero/One Balance Restoration and LLR Compensation

FIG. **3**A provides alternate views of SSD **101**, SSD Controller **100**, and NVM **199** of FIGS. **1**A and **1**B, redrawing selected logic block boundaries to emphasize particular functions. SSD **101** at the top level features External Interfaces **110**, SSD Controller **100**, NVM **199**, and Device Interface **190**.

External Interfaces **110** is coupled to a host, such as Host **102** of FIG. **1**B, and supports a high-level storage protocol such as SATA, including host-sourced storage-related commands and write-data and controller-sourced read-data, and as detailed in the above discussion of FIG. **1**A. Device Interface **190** supports low-level NVM I/O transactions, as detailed in the discussion of FIG. **1**A. NVM **199** features NVM Array **332**, Programmable Read Voltage Circuitry **334**, Control/Status Registers **336**, and I/O **338**. SSD Controller **100** features Host Interface **111**, Scrambler **310**, ECC Encoder **320**, Device Interface Logic **191**, Imbalance Detector **340**, Buffer **131**, Soft-Decision Capable ECC Decoder **360**, Descrambler **370**, LLR LUT/Selector **350**, and Distribution Tracking, Soft-Decision Bias-Reduction/Compensation Logic **380**. Host Interface **111** couples with the Host via External Interface **110** as discussed previously, provides Write Data **306** to Scrambler **310**, and receives Read Data **351** from Buffer **131**.

The write path and functionality are as follows. Scrambler **310** operates on Write Data **306** to generate Scrambled Write Data **311** to ECC Encoder **320**. Scrambler **310** scrambles Write Data **306** in a reversible manner and such that the Scrambled Write Data **311** has a known statistical distribution of states stored. In some embodiments, Scrambler **310** performs encryption, such as via an AES encoder, which randomizes the data as a by-product of encryption. In some embodiments, Scrambler **310** uses a Linear Feedback Shift Register (LFSR) to randomize the data (but without any data security intent). ECC Encoder **320** processes the Scrambled Write Data **311** to add additional ECC bits, resulting in ECC Encoded Write Data **321**, which is provided to Device Interface **191** for storing into NVM **199** via Device Interface **190**. Generally, the redundant information added by ECC encoding scrambled data has a 50-50 balance of zeroes and ones.

The basic read path and functionality are as follows. NVM pages, including Uncorrected and Scrambled (Raw) Read Data **359**, are received from NVM **199** via Device Interface **190**. In some embodiments and as detailed in FIG. **3**B, each NVM page read includes four codewords (a.k.a. read units): Codeword 0 through Codeword 3. Continuing with respect to FIG. **3**A, each codeword nominally includes the data scrambled by Scrambler **301** as well as the additional ECC bits generated by ECC Encoder **320**, but as generally unintentionally altered as a result of NVM storage and retrieval operations. The codewords are provided to Soft-Decision Capable ECC Decoder **360** via Buffer **131**. Soft-Decision Capable ECC Decoder **360** processes the codewords, exploiting the included additional ECC bits to generally correct any errors that may have arisen at any point and time since the data was first ECC encoded, resulting in Corrected Read Data **361**, which is provided to Descrambler **370**. Descrambler **370** operates on Corrected Read Data **361**, reversing the scrambling performed by Scrambler **310**, resulting in Unscrambled Read Data **371**, at least a portion of which is provided as Read Data **351** to Host Interfaces **111** via Buffer **131**.

The read path and functionality further includes Imbalance Detector **340**, LLR LUT/Selector **353**, and Distribution Tracking, Soft-Decision Bias Reduction/Compensation Logic **380**. Imbalance Detector **340** monitors Uncorrected and Scrambled (Raw) Read Data **359** received from the NVM, and at least temporarily maintains data on the statistical distributions of state. Distribution Tracking, Soft-Decision Bias Reduction/Compensation Logic **380** interacts with NVM Control/Status Registers **336**, via Device Interface Logic **191** and NVM I/O **338**, to selectively adjust NVM read thresholds (used to read NVM Array **332**) via NVM Programmable Read Voltage Circuitry **334**. The read thresholds in some embodiments are adjusted at least sometimes to perform zero/one balance restoration based upon a) ECC Status **362** detected by Soft-Decision Capable ECC Decoder **360**, and b) Imbalance **341** (in the statistical distributions of state) detected by Imbalance Detector **340**. The read unit is binary data in both SLC and MLC embodiments. Disparities in the zeroes and ones are directly measurable in the SLC case (or when an MLC is operated as though it were an SLC). In some embodiments, disparities in the more than two states of an MLC are inferred based on knowledge of how the data was stored in the NVM.

LLR LUT/Selector **353** provides Selected LLR **354** soft-decision information sets to the Soft-Decision Capable ECC Decoder **360** for processing in conjunction with the codewords provided to the Soft-Decision Capable ECC Decoder **360** via Buffer **131**. LLR LUT/Selector **353** is managed by Distribution Tracking, Soft-Decision Bias Reduction/Compensation Logic **380** via LLR Control **381**. According to embodiment, LLR LUT/Selector **353** acts as a collection of one or more previously calculated LLR sets stored in look-up tables (LUTs), a cache of previously calculated LLR sets previously retrieved from an on-chip or off-chip store, or a selector for multiple LLR sets (Multi-LLR **352**) streamed via Buffer **131**. LLR LUT/Selector **353** selects one of one or more LLR sets, each of the selected sets comprising one LLR entry for each legal pattern of values obtained from one or more reads of the read unit being decoded.

As illustrated in FIG. **3**C, in some embodiments, soft-decision processor **363** converts multiple reads **365**, such as from a plurality of reads at different read threshold voltages, into soft-decision values, such as in the form of LLR values, for processing by soft-decision capable ECC decoder **360**.

The multiple reads are provided to soft-decision processor **363** in a defined order, such as by increasing or decreasing read threshold voltage, and soft-decision processor **363** converts the pattern received for each bit position into one of the soft-decision values, using the selected LLR **354** to map the patterns into the appropriate LLR values. For example, an SLC bit (or an MLC LSB bit) with five reads has the valid patterns (in decreasing read threshold voltage order, assuming the erased state is a logic '1') 11111, 01111, 00111, 00011, 00001, and 00000. With Grey coding, an MLC MSB with five reads has the valid patterns 11111, 01111, 00111, 00011, 00001, and 00000 (if the LSB bit is '1'), or 00000, 10000, 11000, 11100, 10000, and 11111 (if the LSB bit is '0'). Because of Grey coding, the MLC MSB pattern 11111 corresponds to a high likelihood of 1 and the MLC MSB pattern 00000 corresponds to a high likelihood of 0 irrespective of the LSB, and the other patterns are unique. In further embodiments, the MLC LSB bit is also used as an input to soft-decision processor **363** to distinguish at least the two version of the 11111 pattern so the two version of the 11111 pattern can be assigned independent soft-decision values.

In various embodiments, soft-decision capable ECC decoder **360** comprises soft-decision processor **363**. In some embodiments, selected LLR **354** is read from buffer **131** along with multiple reads **365**. In other embodiments, LLR LUT selection **353** is performed in firmware and selected LLR **354** is provided as part of control information sent to soft-decision processor **363**.

FIG. **3**A illustrates functions for LLR channel variance compensation and zero/one balance restoration in a context where a host operates as a source and a sink for data written to and read from NVM **199**. In various embodiments (such

as illustrated by FIG. **1**A and FIG. **1**B), one or more other agents operate as sources and sinks for data written to and read from NVM **199**. An example of the other agents is Recycler **151** of FIG. **1**A, that moves data from one location to another in NVM **199**, as described elsewhere herein.

In various embodiments, any or all portions of functionality associated with any one or more of Scrambler **310**, ECC Encoder **320**, Soft-Decision Capable ECC Decoder **360**, and Descrambler **370** are implemented at least in part using one or more of Data Processing **121** and ECC **161** (both of FIG. **1**A).

Control Flows for Zero/One Balance Restoration

FIG. **4** illustrates selected control flow details for an embodiment of the SSD of FIGS. **1**A, **1**B, and **3**A, providing particular details regarding write related operations and read related operations. Write operations start with action **410** and continue through **430**. While a return path is not explicitly drawn, subsequent write operations start anew with action **410**. Similarly, read operations start with action **440**, continue through **460**, and conditionally continue through **480**. While a return path is not explicitly drawn, subsequent read operations start anew with action **440**. It is assumed that at least an initial write precedes a read to the same location. Otherwise, except for contrived cases, such as in testing, the read operations and write operations are generally invoked independently as application needs dictate, for the same or generally different locations. Nevertheless, in a heuristic example, the first read operation **440** conceptually continues immediately after the last write operation **430** for a same location.

Considering in more detail the write related operations, original data to be written to the NVM is generally first scrambled in action **410**, such as via encryption or use of an LFSR. Except where more specificity is employed, the term "scrambling" as used herein refers to the manipulation of the original data to be written via any manipulation or selection that results in a known statistical distribution among the various states stored in the NVM. Scrambling embodiments include encryption and LFSR techniques. (See also Scrambler **310** of FIG. **3**A.) In some encryption embodiments, the scrambling is in accordance with at least one version of the AES encryptions specification. In some LFSR embodiments, the scrambling is in accordance with at least one choice of LFSR. In some embodiments, the scrambling is more directly determined via the selection of one or more test patterns, providing a chosen statistical distribution of states when written to the NVM (without necessarily using encryption or LFSR processing).

The LFSR is a hardware shift register having a combinational logic feedback network generating an input bit from a linear function of selected taps of the shift register, or a software model thereof. The LFSR generates a deterministic pseudorandom sequence. At least conceptually, the LFSR-generated pseudorandom sequence is modulo 2 added to the original data to provide the scrambled data.

Subsequently, the scrambled data is next optionally ECC encoded, as illustrated in action **420**. (See also ECC Encoder **320** of FIG. **3**A.) ECC encoding provides redundant information that facilitates data recovery (via error correction) in the face of errors that arise for a variety of reasons when using NVM. Such errors are more prevalent, and thus ECC encoding more advisable, with one or more of smaller geometries, greater use, greater age, greater temperature excursions, and use of MLC memories.

The scrambled and optionally ECC encoded data is then written to the NVM, as shown in action **430**, the last of the write related operations. The distribution of states being written is known as a result of the scrambling. In the case of the use of SLC memories and AES encryption, the zero-one distribution is known to be 50-50 (e.g., a 50 percent chance of a given bit being a zero, and a 50 percent percent chance of a given bit being a one). Using 4LC (4-level MLC) flash memories and AES encryption, the distribution after writing both lower and upper pages is known to be 25-25-25-25 (e.g., a 25 percent chance of being in each of the four states).

In some embodiments, at least one reference region is dedicated for the use of one or more test patterns as discussed above. In some embodiments, the location of the reference region is selected to minimize (or conversely, to maximize) the impact on the test patterns of various NVM write/read artifacts. In some embodiments, the reference region is a reference page, and its location is chosen to be a last page written in a block, so as to minimize the impact of write disturbs on the data written. As some of the phenomena affecting the voltage distributions are time and/or temperature variant, the reference region serves as a known measure of these effects on zero/one disparity.

Considering in more detail the read related operations, starting in action **440**, one or more read units are read from the NVM and the distribution of states is captured. In some embodiments, each of a zeroes count and a ones count are directly determined. Next, in action **450**, an attempt is made to recover the original data. This performs the inverse of whatever data manipulation operations were performed prior to writing. Generally, ECC encoding was added prior to writing, and thus generally the read units next undergo ECC decoding. (See also ECC Decoder **360** of FIG. **3**A.) On a per read unit basis, if ECC decoding is successful, then descrambling is performed next, as appropriate. E.g., where AES encryption was used prior to writing, AES-decryption is performed. Or, where an LFSR-generated pseudorandom sequence was used, the same sequence is modulo 2 added to the ECC decoded data from each of the read units.

The read related operations conclude early when the original data is recoverable, either without error or for errors that are correctable via ECC decoding. See the "no" path from decision **460**. However, on uncorrectable errors of any of the read units, actions **470** and **480** are also performed. See the "yes" path from decision **460**. In action **470**, a disparity in the distribution of states is evaluated. (See also Imbalance Detector **340** of FIG. **3**A.) In some embodiments, the determined disparity is a determined difference between the zeroes count and the ones count. In various embodiments, the determined disparity is computed over all of the read units, even if less than all of the read units had uncorrectable errors. In some embodiments, the disparity computed for an MSB read unit of an MLC NVM is according to a state of the corresponding LSB read unit.

Next, in action **480**, an adjusted value (a.k.a. a threshold shift) for at least one read threshold is determined at least in part based on the magnitude of the determined disparity. In some embodiments, if the magnitude of the determined disparity is below a predetermined tolerance, the read threshold is not adjusted. In some embodiments, a multi-factor "correction algorithm" is employed for determining the adjusted read threshold, for which the magnitude of the determined disparity is but one factor.

In some embodiments, the correction algorithm takes the form of a lookup table used to select a replacement value for the read threshold. The magnitude of the determined disparity is the basis for at least part of the index in the lookup table. In some embodiments, the lookup table is pre-characterized (weighted) based on the particular vendor of the NVM being used. In some embodiments, the index has

additional components, including one or more of: P/E cycles, age, retention time (time since the last write), temperature, and any other factors that may pertain to drift of, or disturbances to, the threshold voltages distributions of the NVM. (See also Distribution Tracking, Soft-Decision Bias Reduction/Compensation Logic **380** of FIG. **3**A.)

Here and throughout the detailed description, it should be remembered that in commonly used flash memory micro-architectures, granularity of operations is different; e.g. a read unit is a minimum size for reading, a page is a minimum size for writing (e.g. programming), and a block is a minimum size for erasing. Each block includes a corresponding plurality of pages, and each page includes a corresponding plurality of read units. In various embodiments, an SSD controller reads from and/or writes to one or more flash memories in quanta corresponding to any one or more of: one or more read units, one or more pages, and one or more blocks.

In some embodiments, disparities are evaluated on an entire page basis. In some embodiments, disparities are evaluated on a read unit (or on a codeword) basis. The disparity evaluation performed in conjunction with an uncorrectable error on a read is performed with respect to the same (page, or codeword) location as written in the most recent write that included that same location.

Control Flows for LLR Compensation

FIGS. **5**A and **5**B illustrate selected control flow details for an embodiment of the SSD of FIGS. **1**A, **1**B, **3**A, and **4**, providing particular details regarding LLR calculation and LLR compensation for channel variance. An exemplary process for LLR calculation, pre-calculation, and updates is set forth by FIG. **5**A. The LLR calculations can be performed for normal reads, retry reads, or diagnostic or other system management reads preformed on demand or as a background task. In some embodiments, the LLR calculations are performed in advance of their needed use. Once calculated, they are stored for subsequent (re)use, until they are replaced as the result of an update. Inner and outer processing loops are shown. The inner loop includes actions **510** and **520**, and action (loop test) **530**. The outer loop includes the inner loop plus actions **540** and **550**.

Starting initially from "LLR Pre-Calculate/Update Start **505**", action **510** provides for initial setting or subsequent stepping (as appropriate) of the read threshold voltage, incrementing the loop step count, and performing a normal or retry read (according to circumstance). Action **520** includes calculating and storing an LLR at the present read threshold voltage. Inner loop test **530** evaluates whether actions **510** and **520** are repeated, such as when additional LLRs are to be calculated at other (stepped) read threshold voltages. According to embodiment, the inner loop is repeated for a predetermined or configured loop count, for example, a loop count of 5, or 10.

Once the desired number of LLRs has been calculated, the inner loop is exited and, as represented by action **540**, the overall process waits (or is suspended) in accordance with a predetermined LLR update strategy, or until a read scrub occurs. According to embodiment, the LLR update strategy includes waiting a day, a week, or another predetermined or configured interval. Once action **540** is satisfied, the overall process resumes in action **550**, where the step count is reset in preparation for starting the inner loop anew. The outer loop then completes, returning the control flow to action **510**, and the process repeats. In some embodiments, the process of FIG. **5**A also calculates a default read threshold voltage, a read threshold voltage increment, or other parameters to be used when reading the NVM.

An exemplary process for performing a flash memory read assisted by LLR compensation is set forth by FIG. **5**B. Starting from Read Start **605**, action **610** represents an attempt at performing a normal read and ECC decode from the flash memory. The read threshold voltage generally will have been previously set, but if not, it is optionally set or adjusted as appropriate. According to embodiment, the ECC decode will be performed as a hard-decision decode, or as any of various soft-decision methods (such as those discussed elsewhere throughout herein). Test "Success ?" **620** evaluates the success of the decode attempt of action **610**. If the decode was successful, then the data read from the flash memory was either error-free or ECC-correctable as a result of the single decode attempt of action **610**, and control flow terminates for this flash memory read at Read Decoded **660**.

If the decode attempt of action **610** was not successful, then control flow proceeds to action group **640** and optionally in parallel with action group **630**. Action group **640** includes a loop with actions **642** and tests **644** and **646**. Action **642** includes re-decoding using the LLR soft-decision sets selected according to a re-decoding step count, and incrementing the re-decoding step count. The LLRs are, e.g., pre-calculated. Action group **640** iteratively performs re-decoding until a decoding success is achieved (Yes, from test **644**) or a predetermined or configured loop count (or timeout) is met (No, from test **646**). Loop termination for reasons other than success are reported or logged as represented by terminus **648**. The SSD controller then takes action according to embodiment and configuration, including waiting for action **632** to attempt a read retry or report a loop termination without success, performing further re-decoding, taking an exception, or other predetermined or configured action.

According to embodiment, executed in parallel with action group **640**, optional action group **630** includes a loop with actions **632** and tests **634** and **636**. Action **632** includes setting or stepping (as appropriate) one or more read threshold voltages, performing a retry read and decode, and incrementing a read retry step count. Action group **630** iteratively performs read retries until a decoding success is achieved (Yes, from test **634**) or a predetermined or configured loop count (or timeout) is met (No, from test **636**). Loop termination for reasons other than success are reported or logged as represented by terminus **638**. The SSD controller then takes action according to embodiment and configuration, including waiting for action group **640** to complete its current loop processing, performing further read retries, taking an exception, or other predetermined or configured action.

A success by either of action groups **630** and **640** is communicated to the other, as represented by the crossed dashed arrows, between action **632** and test **644**, and action **642** and test **634**. In this way, subsequent to a decoding success in either action groups, control flow leaves both action groups and terminates for this flash memory read at Read Decoded **660**.

The re-decoding in action **642** is done based on a prior read (whether a normal read or a retry read). In some embodiments, the re-decoding of action **642** occurs at up to 50× the speed of a retry read and decode attempt of action **632**. That is, in some embodiments, action group **640** can perform 50 re-decoding attempts in the span it takes action group **630** to perform one read retry and decode attempt. According to embodiment, subsequent to "new" retry read data, the re-decoding can continue to attempt soft-decision decoding on the prior read data using different LLR sets (corresponding to different read thresholds) up to the pre-

determined or configured count, or the re-decoding can straight away begin re-decoding with the more recent retry read data. Accordingly, in some embodiments, action group **640** is selectively "synchronized" with each retry read implemented by action group **630**, as represented by the dashed arrow between action **632** and action **642**. More particularly, in some embodiments, action **642** is selectively made to wait for the retry read data from action **632**, and/or loop processing by action group **640** is selectively reset/restarted in view of retry read data received by action **642** from action **632**.

Example Implementation Techniques

In some embodiments, various combinations of all or portions of operations performed by an SSD providing soft-decision compensation for channel variation, e.g., with flash memories, a computing-host flash memory controller, and/or an SSD controller (such as SSD Controller **100** of FIG. **1**A), and portions of a processor, microprocessor, system-on-a-chip, application-specific-integrated-circuit, hardware accelerator, or other circuitry providing all or portions of the aforementioned operations, are specified by a specification compatible with processing by a computer system. The specification is in accordance with various descriptions, such as hardware description languages, circuit descriptions, netlist descriptions, mask descriptions, or layout descriptions. Example descriptions include: Verilog, VHDL, SPICE, SPICE variants such as PSpice, IBIS, LEF, DEF, GDS-II, OASIS, or other descriptions. In various embodiments, the processing includes any combination of interpretation, compilation, simulation, and synthesis to produce, to verify, or to specify logic and/or circuitry suitable for inclusion on one or more integrated circuits. Each integrated circuit, according to various embodiments, is designable and/or manufacturable according to a variety of techniques. The techniques include a programmable technique (such as a field or mask programmable gate array integrated circuit), a semi-custom technique (such as a wholly or partially cell-based integrated circuit), and a full-custom technique (such as an integrated circuit that is substantially specialized), any combination thereof, or any other technique compatible with design and/or manufacturing of integrated circuits.

In some embodiments, various combinations of all or portions of operations as described by a computer readable medium having a set of instructions stored therein, are performed by execution and/or interpretation of one or more program instructions, by interpretation and/or compiling of one or more source and/or script language statements, or by execution of binary instructions produced by compiling, translating, and/or interpreting information expressed in programming and/or scripting language statements. The statements are compatible with any standard programming or scripting language (such as C, C++, Fortran, Pascal, Ada, Java, VBscript, and Shell). One or more of the program instructions, the language statements, or the binary instructions, are optionally stored on one or more computer readable storage medium elements. In various embodiments some, all, or various portions of the program instructions are realized as one or more functions, routines, sub-routines, in-line routines, procedures, macros, or portions thereof.

CONCLUSION

Certain choices have been made in the description merely for convenience in preparing the text and drawings and unless there is an indication to the contrary the choices should not be construed per se as conveying additional information regarding structure or operation of the embodiments described. Examples of the choices include: the particular organization or assignment of the designations used for the figure numbering and the particular organization or assignment of the element identifiers (the callouts or numerical designators, e.g.) used to identify and reference the features and elements of the embodiments.

The words "includes" or "including" are specifically intended to be construed as abstractions describing logical sets of open-ended scope and are not meant to convey physical containment unless explicitly followed by the word "within."

Although the foregoing embodiments have been described in some detail for purposes of clarity of description and understanding, the invention is not limited to the details provided. There are many embodiments of the invention. The disclosed embodiments are exemplary and not restrictive.

It will be understood that many variations in construction, arrangement, and use are possible consistent with the description, and are within the scope of the claims of the issued patent. For example, interconnect and function-unit bit-widths, clock speeds, and the type of technology used are variable according to various embodiments in each component block. The names given to interconnect and logic are merely exemplary, and should not be construed as limiting the concepts described. The order and arrangement of flowchart and flow diagram process, action, and function elements are variable according to various embodiments. Also, unless specifically stated to the contrary, value ranges specified, maximum and minimum values used, or other particular specifications (such as flash memory technology types; and the number of entries or stages in registers and buffers), are merely those of the described embodiments, are expected to track improvements and changes in implementation technology, and should not be construed as limitations.

Functionally equivalent techniques known in the art are employable instead of those described to implement various components, sub-systems, operations, functions, routines, sub-routines, in-line routines, procedures, macros, or portions thereof. It is also understood that many functional aspects of embodiments are realizable selectively in either hardware (e.g., generally dedicated circuitry) or software (e.g., via some manner of programmed controller or processor), as a function of embodiment dependent design constraints and technology trends of faster processing (facilitating migration of functions previously in hardware into software) and higher integration density (facilitating migration of functions previously in software into hardware). Specific variations in various embodiments include, but are not limited to: differences in partitioning; different form factors and configurations; use of different operating systems and other system software; use of different interface standards, network protocols, or communication links; and other variations to be expected when implementing the concepts described herein in accordance with the unique engineering and business constraints of a particular application.

The embodiments have been described with detail and environmental context well beyond that required for a minimal implementation of many aspects of the embodiments described. Those of ordinary skill in the art will recognize that some embodiments omit disclosed components or features without altering the basic cooperation among the remaining elements. It is thus understood that much of the details disclosed are not required to implement various aspects of the embodiments described. To the extent

that the remaining elements are distinguishable from the prior art, components and features that are omitted are not limiting on the concepts described herein.

All such variations in design are insubstantial changes over the teachings conveyed by the described embodiments. It is also understood that the embodiments described herein have broad applicability to other computing and networking applications, and are not limited to the particular application or industry of the described embodiments. The invention is thus to be construed as including all possible modifications and variations encompassed within the scope of the claims of the issued patent.

What is claimed is:

1. A method, comprising:

performing an initial read and an initial attempted ECC soft-decision decoding, for each of a plurality of flash memory locations;

performing up to a first plurality of subsequent ECC soft-decision redecoding attempts based on the initial read, subsequent to failure of the initial attempted ECC decoding, for at least some of the locations;

wherein each initial attempted ECC soft-decision decoding and each subsequent ECC soft-decision redecoding attempt is performed at least in part using a controller of a solid-state disk; and

wherein each ECC soft-decision redecoding attempt uses a selected one of a plurality of pre-calculated soft log likelihood ratio (LLR) soft-decision information sets, and wherein each of the pre-calculated soft LLR soft-decision information sets corresponds to a respective voltage distribution.

2. The method of claim **1**, wherein at least some of the selected LLR soft-decision information sets are selected in accordance with a read threshold-tracking algorithm and a predetermined retry strategy.

3. The method of claim **2**, wherein the threshold-tracking algorithm comprises evaluating a zero-one balance for each of a corresponding plurality of reads.

4. The method of claim **3**, wherein the initial LLR soft-decision information set corresponds to an initial read threshold voltage predicted to be associated with a lowest disparity of the zero-one balances evaluated.

5. The method of claim **4**, further comprising selecting subsequent LLR soft-decision information sets corresponding to read threshold voltages alternating on either side of the initial read threshold voltage in incrementally increasing amounts.

6. The method of claim **5**, further comprising at least sometimes updating at least some of the LLR soft-decision information sets on a periodic basis.

7. The method of claim **6**, wherein:

the LLR soft-decision information sets comprise a first one of a plurality of LLR soft-decision information groups; and

at least a portion of the flash memory is organized into a plurality of redundancy groups, each of the redundancy groups being associated with a corresponding one of the LLR soft-decision information groups.

8. The method of claim **7**, wherein at least some of the LLR soft-decision information groups used for soft-decision-decoding and soft-decision-redecoding are specific to each of a plurality of redundancy groups in which the flash memory is organized.

9. The method of claim **8**, further comprising streaming data from each read and at least some of the redundancy-group-specific corresponding LLR soft-decision information groups to a soft-decision processing unit to produce soft-decision information, wherein the redecoding attempts are performed on the soft-decision information.

10. A device, comprising:

means for performing an initial read and an initial attempted ECC decoding, for each of a plurality of flash memory locations;

means for performing up to a first plurality of subsequent ECC soft-decision redecoding attempts based on the initial read, subsequent to failure of the initial attempted ECC decoding, for at least some of the locations;

wherein each initial attempted ECC soft-decision decoding and each subsequent ECC soft-decision redecoding attempt is performed at least in part using a controller of a solid-state disk; and

wherein each ECC soft-decision redecoding attempt uses a selected one of a plurality of pre-calculated soft log likelihood ratio (LLR) soft-decision information sets, and wherein each of the pre-calculated soft LLR soft-decision information sets corresponds to a respective voltage distribution.

11. The device claim **10**, further comprising means for performing a second plurality of retry reads and respective retry ECC decoding attempts, overlapped in time with the first plurality of ECC soft-decision redecoding attempts, for the at least some of the locations.

12. The device of claim **11**, wherein at least some of the selected LLR soft-decision information sets are selected in accordance with a read threshold-tracking algorithm and a predetermined retry strategy.

13. The device of claim **12**, wherein the threshold-tracking algorithm comprises evaluating a zero-one balance for each of a corresponding plurality of reads.

14. The device of claim **13**, wherein the initial LLR soft-decision information set corresponds to an initial read threshold voltage predicted to be associated with a lowest disparity of the zero-one balances evaluated.

15. The device of claim **14**, further comprising means for selecting subsequent LLR soft-decision information sets corresponding to read threshold voltages alternating on either side of the initial read threshold voltage in incrementally increasing amounts.

16. The device of claim **15**, further comprising means for at least sometimes updating at least some of the LLR soft-decision information sets on a periodic basis.

17. The device of claim **16**, wherein:

the LLR soft-decision information sets comprise a first one of a plurality of LLR soft-decision information groups; and

at least a portion of the flash memory is organized into a plurality of redundancy groups, each of the redundancy groups being associated with a corresponding one of the LLR soft-decision information groups.

18. The device of claim **17**, wherein at least some of the LLR soft-decision information groups used for soft-decision-decoding and soft-decision-redecoding are specific to each of a plurality of redundancy groups in which the flash memory is organized.

19. The device of claim **17**, further comprising streaming data from each read and at least some of the redundancy-group-specific corresponding LLR soft-decision information groups to a soft-decision processing unit to produce soft-decision information, wherein the redecoding attempts are performed on the soft-decision information.

20. A solid-state disk controller, comprising:

a buffer memory;

host interface logic configured to interface with a host computing system;

device interface logic configured to interface with one or more flash memory devices; and

a processor configured to perform a method comprising:

performing an initial read and an initial attempted ECC decoding, for each of a plurality of flash memory locations;

performing up to a first plurality of subsequent ECC soft-decision redecoding attempts based on the initial read, subsequent to failure of the initial attempted ECC decoding, for at least some of the locations;

wherein each ECC soft-decision redecoding attempt uses a selected one of a plurality of pre-calculated soft log likelihood ratio (LLR) soft-decision information sets, and wherein each of the pre-calculated soft LLR soft-decision information sets corresponds to a respective voltage distribution.

* * * * *